Global Information Assurance Certification Paper

Auditing RedHat Linux 7.0

(workstation)

Mary A. Laude

submitted on 23 July, 2001 for
GSNA (GIAC Systems and Network Auditor) certification
as GSNA Practical Assignment, Version 1.0

Introduction

The purpose of this paper is to present a baseline audit of RedHat Linux 7.0 Workstation on an x86 architecture. A good audit comprises many detailed steps, and I hope to present them clearly.

Ideally, a baseline audit should include hardening the system and should show the state of the system when it is compliant with the security audit policy.

The security policy will probably exist in some form before the audit takes place; it should be reviewed and updated prior to the audit. If the policy does not exist then the auditor should construct one. Research into the purpose, usage, availability, maintenance, and physical security of the system should help decide the policy.

For the purpose of this paper, let's assume that the system is a desktop workstation on a corporate intranet. It should not be running a web server and should not be sharing files with other systems. It uses a remote printer and communicates with an internal email server in order to send and receive email. It needs access to the internet.

The security policy for this host is the following:

- Services that need to be running and what ports they should be using are the following:
  - ssh, port 22
- User accounts that should be on the system are root, sys, john, betty, mary.
- Password policy: every user must have a password and it must be at least 7 characters long with at least one alpha and one special/digit included.
- Remote users john, betty, and mary are allowed to use the system via ssh.
- No remote login is allowed for root or sys.
- Users root and john are allowed to login at the console.
- The system should not be sharing files (no file systems should be exported).
- The system should not be mounting remote file systems (no nfs mounts).
- The system should not be running either a web or an ftp server.
- The system should be protected with anti-virus software.
- The state of the system should be up and running, with automatic screensaver and lock if unattended for 5 minutes.
- The physical location of the system is John's cube.

The hostname for this host to be audited is **audithost**.

My audit will define a security audit policy, using the general security policy as a minimal set of constraints. The security audit will concentrate on the hardening of this system, to bring it up to policy, and then will suggest tools to put in place in order to automate subsequent audits. The audit will also suggest additions to the security policy regarding logging, file integrity checks, and immutable files.

In my examples below, command line prompts are usually the dollar sign ($); this should not be taken to mean that all of these commands should be able to be executed by a non-privileged user. The dollar sign prompt is presented for convenience and consistency. Screen shot examples are shown at the end of the paper in order to eliminate wasted space in the text.

Note: While every effort has been made to make sure all of the URLs referenced in this paper work, there is no way for me to know if they will still be working by the time you read this. I apologize in advance for any broken links.

The Current State of Affairs

Current Checklists

Currently, I cannot find a checklist specifically for auditing RedHat 7.0 workstation, however the following publications are very good references.

- The SANS Institute. Securing Linux Step-by-Step. Copyright 1999, 2000.
- Naidu, Krishni. The SANS Institute. "Auditing Linux". Checklist provided to Auditing Networks track attendees at SANS Baltimore, May 2001. URL: http://www.sans.org/checklist/linux_check.htm.
- Whelan, Paul. The SANS Institute. "Linux Security Auditing". June 1, 2001. URL: http://www.sans.org/infosecFAQ/audit/linux_sec.htm.

I particularly enjoyed reading Paul Whelan's paper. It was a clear and concise commentary, and fairly up to date on current Linux security practice. The SANS Step-by-Step was a very thorough look at RedHat 6.0. A striking difference between RedHat 6.0 and 7.0 is the structure of the inetd configuration files; no longer is there just one file, inetd.conf. In 7.0, there is an xinetd.conf file, which includes the directory /etc/xinetd.d, which is the directory which contains a configuration file for each service. Oh, and the service is called xinetd.

```
$ cat /etc/xinetd.conf
#
# Simple configuration file for xinetd
#
# Some defaults, and include /etc/xinetd.d/

defaults
{
        instances               = 60
    log_type                = SYSLOG authpriv
```

```
        log_on_success          = HOST PID
        log_on_failure          = HOST RECORD
}

includedir /etc/xinetd.d
```

There are various other online references which detail steps for securing the Linux operating system, and I have listed them in the References section of this paper. There are also countless checklists for auditing the UNIX operating system. Though general in scope, some of the items in these checklists can be modified and made items in a RedHat 7.0 workstation audit checklist.

There are lots of theories and wise commentaries available regarding the topic of hardening Linux. Part of the mission of a good audit is to harden the system, according to policy. Once the system is brought up to policy standards, it is the regular automated audit which will find when a system falls out of policy.

Current Tools

Krishni Naidu, author of "Auditing Linux" mentions Tiger, TARA, SATAN, SARA, and SAINT for performing vulnerability analysis, Tripwire for monitoring file modifications, and Psionic Port Sentry for host-based IDS.

The SANS Institute, in their publication entitled Securing Linux Step-by-Step, mentions Swatch and Psionic Logcheck for log monitoring, Tripwire for file mod monitoring, Psionic Port Sentry for host-based IDS, Tiger and TARA for host-based vulnerability analysis, and SARA, SAINT, Nessus, Nmap, ISS Internet Scanner, and Cybercop for network-based vulnerability analysis.

Paul Whelan, author of "Linux Security Auditing", mentions Swatch and Logwatch for log monitoring, Tripwire and AIDE for file integrity checking, and Snort and Tcpdump for traffic sniffing.

It should be noted that in each checklist, tools are mentioned and explained, with the idea that the auditor will choose from the large set the tools that he or she feels are the best subset for the particular system or systems being audited. It would not be necessary or even practical for an audit to use every one of the tools mentioned.

Improvements Needed

Papers which discuss audit and hardening of systems often forget to mention two items regarding procedure. One is that the auditor should always be using tools that he or she brings to the audit, not tools that are already installed. This includes system utilities. The reason for this is, as most of us know, you never know where they've been. You don't want to risk using a tampered-with utility to provide you with reliable system state.

The other procedural note is that while you are conducting the audit, it's a good idea to disconnect the system from the internet. If you need to have communication with another system, for doing a nessus scan for example, you can configure that communication exclusively.

The reason for this is that you don't know who's watching when you are finding out all the details of the system via your audit.

I have found that sometimes methods are missing from checklists. It is not sufficient, in my opinion, to write a checklist of items which include only a statement of the check that is being done and not the method to achieve the check. For example, a check to make sure that "shadow passwords with MD5 hashing" are in use should include the method for how to find out if they are. If you didn't install the operating system yourself, how do you know? To take away any doubt, let's include methods.

Output from some tools can be misleading, too general, or even incorrect. Sometimes this is because the versions of the tools available are not in sync with the versions of operating systems we are using them for. This can lead to a confused state of results.

For example, a run of nmap with the -O option includes a check of the operating system. Included in the output should be the version of the Linux kernel that is running. In my audit, I was using version 2.53 of nmap and running it on RedHat 7.0. The kernel version that nmap reported was incorrect. (Well, it does say that it's a guess…)

> nmap: Remote operating system guess: Linux 2.1.122 - 2.2.14
>
> uname -a: Linux audithost 2.2.16-22 #1 Tue Aug 22 16:16:55 EDT 2000 i586 unknown

Most checklists start with the install of the Linux box. They cite procedures that one should follow in order to install and configure the operating system, from defining partitions to choosing packages to configuring the kernel. This advice is great. However, in the field, the auditor frequently finds that she must audit a system which is already up and running and being used on a day-to-day basis. Applications have already been installed on top of the operating system, and they have been configured by the owner of the box. The auditor's task is to audit without disturbing them (the applications or the owner).

I have not found a checklist which includes a check for anti-virus software, yet it seems to me that every workstation should have this installed. Right? Also, none of the checklists I consulted dealt with the issue of physical security, though this is an important consideration.

My formula for a practical audit is to put together a checklist that does not make any assumptions about how the system was installed, and takes into account the fact that applications are already running (that need to keep running, if possible).

For the case of RedHat Linux 7.0 workstation, the existing battery of checklists can be sorted through and combined to come up with a more precise checklist. I will present my checklist in two parts: objective items and subjective items. I will also present the list of tools in my toolkit.

My Toolkit

For my audit, I am relying entirely on tools that either are bundled with the operating system or are available over the internet as freeware. There are a couple of reasons for this. First, there are a number of really cool tools that do the job well (if you use them correctly for your situation,

and you are aware of the caveats of each) and are also free. Second, I have no job at present so have neither access to systems with commercial tools on them nor money to spend.

Let me reiterate the importance of keeping an auditor's toolkit on an external medium (like a cdrom), so that the auditor is assured of the integrity of the tools. This includes the command line utilities used. If you didn't install the operating system yourself, how can you be sure the system utilities haven't been compromised?

I think it's a good idea to remind the auditor that nobody is infallible. If you write a script, be sure to test it before you run it on a system you are auditing. Everyone makes mistakes, but you don't want to compromise your audit. If you have another pair of eyes look at your script and run it, you can be more certain of the reliability and robustness of your script.

The following is a list of tools that I will use for this audit. Exact reference information for each tool can be found in the Tools Reference List at the end of this paper.

- command line utilities
  - give diagnostic information about the state of the system
  - e.g. uname, netstat, find, ls, etc.
- nmap
  - network mapping tool, gives info about open ports and services running
  - extremely easy tool to use
- nessus
  - vulnerability scanner, can run nmap plus try various attacks
  - I built a nessus server on Solaris, then a Windows client from where I scanned.
- lsof
  - lists open network sockets, useful for knowing
  - gives good information; interpreting it is challenging
- Bastille
  - system hardening script
  - useful for implementing some of the recommendations in the results of an audit
- Swatch
  - a simple log watcher
  - challenging to figure out how to make it work on Red Hat 7
- COPS
  - Computer Oracle and Password System, an auditing script
  - I downloaded this but decided not to use it – it tells what the vulnerabilities are; it doesn't fix them.
- Tripwire
  - File Integrity Assessment tool
  - Fairly easy to install, configure and run; gives a nice report.

My Objective Checklist

It is important to have objective measurements during an audit. Objective measurements are important because they are not dependent on the auditor's method, mood, or judgement.

The following is my checklist of objective measurements. Note that even though we are auditing RedHat 7.0 Workstation, since we did not install the operating system ourselves, we are assuming nothing about services. The outcome of the audit (including hardening) will be a system which is a workstation, not a server. This checklist is terse and nit-picky; however, I wanted to be thorough.

It is important to note that when using the various tools to determine the objective measurements, results from tools can disagree. This is why I've included redundant checks. For example, there are several tools that can be used to answer the question "what services are running?" It is important to make sure you get the best-informed answer.

General

- What is the version of the kernel? Run **uname –a**. Pass: Output should show 2.2.16-22 for default 7.0 install; 2.2.19-7.0.1 after security updates have been installed. Compare this result with results from nessus; nessus output may show an incorrect kernel version, depending on which version of nessus you are using.

  ```
  $ uname -a
  Linux audithost 2.2.16-22 #1 Tue Aug 22 16:16:55 EDT 2000 i386 unknown
  ```

- The operating system has the latest security patches/updates. Pass: Yes, the versions of applications from the output of **rpm –qa** match the versions of the applications which contain the latest security updates. See Appendix A for the versions of packages which should be installed according to http://www.redhat.com/support/errata/rh7-errata-security.html .
- Is there a recovery boot disk? Pass: Yes, a recovery boot disk has been made after system is hardened. Use /sbin/mkbootdisk.

User Accounts

It should be noted that I define a privileged user as one that is in group 0, the privileged group. By default the following users are privileged: root, sync, shutdown, halt, operator. And by default, the users root and operator should be the only users from this group who are allowed to have interactive shells.

- The only non-privileged user accounts are the accounts listed in the security policy. Pass: Yes, examination of **/etc/passwd** reveals the only non-privileged accounts are john, betty, and mary.
- Shadow passwords with md5 hashing are enabled. Pass: Yes, the file /etc/shadow exists, it is mode 600, and an md5 hash performed on a particular password yields the same hash as the corresponding entry in the /etc/shadow file.
- There is a password for every user and every user's password is shadowed. Use **pwck** to verify passwd and shadow files. Pass: Yes, **pwck** output shows no errors.
- The system password policy matches the security policy. Pass: Yes, *linuxconf* reflects password constraints from security policy: Examine User Accounts => Policies => Password & Account Policies screen of *linuxconf*.

System Configuration

- There is anti-virus software installed. Pass: Yes, there is anti-virus software installed and running.

- There are no shared filesystems. Pass: Yes, there are no export filesystems configured in **/etc/exports** and the command **exportfs** produces no output.

```
$ ls -l /etc/exp*
-rw-r--r-- 1 root root 0 Jan 12 2000 /etc/exports
$ cat /etc/exports
$ exportfs
$
```

- There are no nfs mounted filesystems. Pass: Yes, the command **df -k** reveals no nfs mounted filesystems.

```
$ df -k
Filesystem 1k-blocks Used Available Use% Mounted on
/dev/hda2  2411408  1021416 1267500 45%  /
```

- There are no setuid or setgid files on the system. Pass: Yes, the commands **find / -perm -4000 -print** and **find / -perm -2000 -print** do not produce any output.
- Is the LILO prompt password protected? (This is important if you want the extra security of needing an additional password besides the root password in order to enter single user mode. Be very careful, however, because this is a kernel file; if your syntax is off or the file gets messed up you will probably not be able to boot. This would, of course, be very bad indeed!) Pass: Yes, the two directives following the prompt directive shown below appear in **/etc/lilo.conf** (where zzzzzzz is a good password).

```
$ cat /etc/lilo.conf
. . .
prompt
password = zzzzzzz
restricted
. . .
```

- Are the file permissions of /etc/lilo.conf 600 (rw by owner only) and is the file owned by root? Pass: Yes, **ls –l /etc/lilo.conf** output shows -rw------ and owned by root.
- Does /etc/inittab show that reboot from console with Ctrl+Alt+Del is disabled? Pass: Yes, the directive allowing this (the default ca directive) has been commented out.

```
# Trap CTRL-ALT-DELETE
#ca::ctrlaltdel:/sbin/shutdown -t3 -r now
```

- Does /etc/inittab show that the root password is required to enter single user mode? Pass: Yes, the *wait* directive has been added below the *sysinit* directive.

```
# System initialization.
si::sysinit:/etc/rc.d/rc.sysinit
~~:S:wait:/sbin/sulogin
```

- Does /etc/security/access.conf show that all console logins except for root and john are disabled? Pass: The *All EXCEPT* directive should be configured to include root and john (users who are allowed to login at the console).

```
# Disallow console logins to all but a few accounts.
#
-:ALL EXCEPT root john:console
```

- Are TCP wrappers in use to manage access to network services? Pass: Yes, the last line in ***/etc/hosts.allow*** is **deny all** and the only uncommented line in ***/etc/hosts.deny*** is **ALL: ALL**

```
#
# hosts.deny  This file describes the names of the hosts which are
#             *not* allowed to use the local INET services, as decided
#             by the '/usr/sbin/tcpd' server.
#
ALL:ALL

#
# hosts.allow This file describes the names of the hosts which are
#             allowed to use the local INET services, as decided
#             by the '/usr/sbin/tcpd' server.
#
all:all:deny
```

- Does /etc/securetty show that remote (telnet) users are not allowed? Pass: Yes, **/etc/securetty** shows file only contains *tty* entries..

Services/Ports

- From the inside: what ports are open? Run **nmap -I -O -sR localhost**. Pass: port 22 for ssh; output should look like the following:

```
$ nmap -I -O -sR localhost

Starting nmap V. 2.53 by fyodor@insecure.org ( www.insecure.org/nmap/ )
Interesting ports on localhost.localdomain (127.0.0.1):
(The 1510 ports scanned but not shown below are in state: closed)
Port       State      Service (RPC)          Owner
22/tcp     open       ssh                    root

TCP Sequence Prediction: Class=random positive increments
                         Difficulty=2276003 (Good luck!)
```

- From the inside: what services are running? Run **netstat -at**. Pass: ssh should be the only service in LISTEN mode.

```
$ netstat -at
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 *:ssh                   *:*                     LISTEN
```

- List all open network sockets. Are there any open sockets that you should be concerned about? Use **lsof –i +M**. Pass: ssh. Browser files are also acceptable. (See below.)

```
$ lsof -i +M
COMMAND    PID USER   FD   TYPE DEVICE SIZE NODE NAME
sshd       537 root    3u  IPv4    686       TCP *:ssh (LISTEN)
netscape- 2549  mary   23u  IPv4  11926       TCP audithost:4055->vic.cc.purdue.edu:ftp (CLOSE_WAIT)
```

- From the outside: do a port scan. Which ports are reported open? Use **nessus**. In the nessus output labeled Nessus Example 1, we can see a typical first pass result. It shows the ports that are open on the target audit system. (Many of these need to be closed.) Pass: only port 22 is visible.

- From the outside: do a vulnerability assessment. Which services are vulnerable? Use **nessus** and look up cve numbers in the cve database at URL: http://cve.mitre.org/cve/refs/refmap/source-REDHAT.html. In the nessus output labeled Nessus Example 2, we see the possible vulnerability of the version of openssh. Pass: the nessus output reveals no vulnerabilities; the ssh version is up to date as far as security releases are concerned.
- Secure remote shell, sshd is running. Pass: **ps –ef | grep sshd** succeeds. (Yes, we already saw that the port is open corresponding to this service but let's check that the daemon is running locally.)
- What is the version of ssh? Pass: **ssh –V** or **rpm –qa | grep ssh** shows ssh 1.2.32 or later, or openssh-2.5.2p2-1.7.2 or later. (At the time of this writing, these are the latest versions of ssh and openssh, respectively. Make sure you have the latest versions which include fixes for known vulnerabilities – see Appendix A.) Compare this result with results reported by nessus, for example. The output below shows that the system needs an updated version.
- Are unneeded services unconfigured? Use **chkconfig --list** to show, for each run level, which services are configured and whether they are *on* or *off*. Pass: this output must be examined closely to determine that only necessary services are, and that of those, they are enabled (on) only at the correct run levels (0-6).

```
$ chkconfig --list
anacron 0:off   1:off   2:on    3:on    4:on    5:on    6:off
. . .
netfs   0:off   1:off   2:off   3:on    4:on    5:on    6:off
network 0:off   1:off   2:on    3:on    4:on    5:on    6:off
. . .
linuxconf 0:off 1:off   2:on    3:on    4:on    5:on    6:off
sshd    0:off   1:off   2:on    3:on    4:on    5:on    6:off
xinetd  0:off   1:off   2:off   3:on    4:on    5:on    6:off
xinetd based services:
                linuxconf-web:  off
```

- Have unneeded services been shut down properly? To shut down a service, use its init script in the **/etc/rc.d/init.d** directory. For example, **/etc/rc.d/init.d/httpd stop**

  DNS: This service has been stopped, unconfigured and remove from the system.
  - There is no DNS server running. Pass: Yes, **ps –ef | grep named** fails. If named is running, execute the following as root:

    ```
    # /etc/rc.d/init.d/named stop
    ```

  - The naming service software been removed from the box. Pass: Yes, rpm -qa does not show *bind* or *caching-nameserver* packages. If packages are still on the box, execute the following as root:

    ```
    # rpm -e caching-nameserver
    # rpm -e bind
    ```

  - Remote nameservers have been configured. Pass: Yes, /etc/resolv.conf defines the IP addresses of primary and secondary nameservers.

    ```
    $ cat /etc/resolv.conf
    nameserver primary_nameserver_IP
    nameserver seconday_nameserver_IP
    ```

- FTP and TFTP: These services are not running, are unconfigured and the software has been uninstalled (removed).
  - There is no FTP service running. Pass: Yes, **netstat -at | grep ftp** does not produce output.
  - The file /etc/ftpusers shows that root and system accounts cannot use ftp. Pass: Yes, contents of /etc/ftpusers does not include root or any privileged users (users who are not allowed to use ftp -- funny the file is not called ftpnonusers, but well…) :

    root
    bin
    daemon
    adm
    . . .
    nobody

  - The file /etc/ftpaccess does not allow guest or anonymous connections (even though you have turned off the ftp service). Pass: Yes, user types have been removed from the class directive.
  - The following configuration files have been removed from the /etc/inetd.d directory: tftp, wu-ftpd. Pass: Yes, **ls /etc/xinetd.d/*ftp*** produces no output.
  - The ftp packages have been removed from the system. Pass: Yes, **rpm –qa | grep ftp** does not produce output. If this check doesn't pass, execute the following as root (where *package* is the name of each ftp package found in the rpm query):

    # rpm –e *package*

- TELNET, FINGER, TALK, RLOGIN, RSH, REXEC (are trouble!): These services are not running, are unconfigured and the software has been uninstalled (removed). For each of these services, check the following (substitute the name of the service):
  - These services are not running. Pass: Yes, **netstat -at | grep *service*** does not produce output.
  - The following configuration files have been removed from the /etc/inetd.d directory: finger, ntalk, rexec, rlogin, rsh, talk, telnet. Pass: Yes, **ls /etc/xinetd.d/** does not show any of these configuration files.
  - The packages corresponding to these services have been removed from the system. Pass: Yes, **rpm –qa | grep *service*** does not produce output. If this check doesn't pass, execute the following as root (where *package* is the name of each package found in the rpm query):

    # rpm –e *package*

- SENDMAIL
  - There is no email server or relay daemon running. Pass: Yes, In /etc/sysconfig/sendmail, the value for DAEMON is no:
    DAEMON=no
  - SMTP vrfy and expn commands cannot be run remotely. Pass: Yes, /etc/sendmail.cf contains the line **PrivacyOptions=goaway**.

- OTHER
  - NFS is not running and the NFS packages are not installed on the system. Pass: Yes**, ps –ef | grep nfs** and **rpm -qa | grep knfsd** do not produce output. If this check doesn't pass, execute the following as root:

    ```
    # /etc/rc.d/init.d/nfs stop
    # rpm -e knfsd
    ```

  - No print server is running. Pass: Yes, **ps –ef | grep lpd** produces no output and **/etc/hosts.lpd**, if it exists, is blank. (Note, this system should configure its /etc/printcap via printtool, to point to a remote print server.)

  - There is no web server running and the web server packages do not exist on the box. Pass: Yes, **ps –ef | grep httpd** and **rpm -qa | grep apache** do not produce output. If this check doesn't pass, execute the following as root:

    ```
    # /etc/rc.d/init.d/httpd stop
    # rpm –e apache
    # rpm –e apache-devel
    ```

- If there are no xinetd services running, xinetd is turned off. Pass: If no xinetd services are running, **ps –ef | grep xinetd** produces no output.

- Have unneeded services been eliminated? After shutting down an unneeded service, remove the init script from the runlevel directory. Use **/sbin/chkconfig** to remove links from the run level directory. For example, **/sbin/chkconfig httpd off.** Pass: This has been done for each service you want turned off.

File Integrity

- There are no **.rhosts** files on the system, in the root file system, or in any user partitions. Pass: Yes, **find / -name .rhosts -print** does not find anything.
- There is no **/etc/hosts.equiv** file. Pass: Yes, **ls –l /etc/hosts.equiv** fails.

- Is Tripwire installed and configured? Pass: Yes, Tripwire has been installed and configured.

Physical Security

- Is the system up and is it protected by an automatic screensaver and lock which starts after the system is idle for 5 minutes? Pass: Yes, when idle for 5 minutes, the system turns on a screensaver and locks itself. The password of the console-logged-in user is required to unlock.
- Is the system in John's cube? Pass: Yes, the system is located in John's cube.

My Subjective Checklist

- There are no weak passwords on the system. Use a password cracker. Pass: The password cracker reports no weak passwords. (This is subjective because the password cracker's idea of what is weak is relative.)
- Is an adequate job being done of writing to logs? Pass: Examine /etc/syslog.conf. List files in /var/log. Questions to answer: What is being logged? How often is it logged? How long are logs kept? Are logs backed up? Are you comfortable with all the answers?

- Is an adequate job being done to monitor logs? Pass: Examine swatch configuration and output. Questions to answer: Is swatch configured and running? Which logs is it monitoring? What actions, if any, are taken?
- Would it make sense to automate the operating system security update process? Pass: Examine how often security updates are released. Questions to answer: Is there ever a case when you wouldn't want to automatically apply a security update? Can you justify the overhead involved with automatic updates? (Perhaps the overhead would be negligible.)
- Is an adequate job being done regarding backups? Pass: What is being backed up? How often? Where are backups kept? Has the restore procedure been tested? Are you comfortable with the answers to these questions?

The Audit

The audit I carried out consisted of the following steps:

1. Audit using the checklist (objective and subjective).
2. Analysis to determine the reason for each audit failure.
3. Recommendation for how to bring each item up to policy (to pass the audit), and the steps and procedures for carrying out each recommendation. (Please note that I am not mentioning the need to reboot the system at every turn; reboot as necessary. After applying some package updates, it will be essential to reboot and rpm should state this, where applicable.)

Before carrying out the recommendations, I installed and ran Bastille, in order to handle a few of these recommendations. It:

- disabled setuid status for mount, umount, ping.
- put password protection on single-user mode.
- put password protection on the LILO prompt and disabled CTRL-ALT-DEL rebooting.
- configured TCP wrappers
- edited the telnet configuration file so that the service default status was *off*. However, it did not restart xinetd (so the services in question were still *on*). Since there is no –hup option for this version of xinetd, I had to kill xinetd and then start it again. Then I removed the telnet conf file and removed the telnet package from the system.
- disabled the print server.
- reconfigured sendmail so that it is in non-daemon mode and so that SMTP vrfy and expn commands cannot be run remotely.

Objective items (a subset)

General

- What is the version of the kernel? Run **uname –a**. To Pass: Output should show 2.2.16-22 for default 7.0 install; 2.2.19-7.0.1 after security updates have been installed.

```
$ uname –a
Linux audithost 2.2.16-22 #1 Tue Aug 22 16:16:55 EDT 2000 i386 unknown
```

Analysis: This check **Failed** since the kernel is at the level of a 7.0 install, instead of the level of the latest security update kernel. Recommendation: Upgrade the kernel to 2.2.19-7.0.1.

1. Download the upgrade from ftp://updates.redhat.com/7.0/en/os/i386/kernel-2.2.19-7.0.1.i386.rpm

2. Install the upgrade by executing, as root, the following command:
   # rpm -Fvh kernel-2.2.19-7.0.1.i386.rpm

- The operating system has the latest security patches/updates. To Pass: Yes, the versions of applications from the output of **rpm -qa** match the versions of the applications which contain the latest security updates. (Note: output from this command is not shown here because it would span several pages.) Analysis: This check **Failed** because there are several security updates which haven't been installed. Recommendation: Install the security updates by downloading them from the list below and using the Freshening option of rpm to upgrade all of the packages at once using the following command: **rpm -Fvh *.rpm**

  existing package: update

  bind-utils-8.2.2_P5-25: ftp://updates.redhat.com/7.0/en/os/i386/bind-utils-8.2.3-1.i386.rpm
  cyrus-sasl-1.5.24-6: ftp://updates.redhat.com/7.0/en/os/i386/cyrus-sasl-1.5.24-11.i386.rpm
  ed-0.2-17: ftp://updates.redhat.com/7.0/en/os/i386/ed-0.2-19.i386.rpm
  esound-0.2.19-3: ftp://updates.redhat.com/7.0/en/os/i386/esound-0.2.20-1.i386.rpm
  esound-devel-0.2.19-3: ftp://updates.redhat.com/7.0/en/os/i386/esound-devel-0.2.20-1.i386.rpm
  gftp-2.0.7b-2: ftp://updates.redhat.com/7.0/en/os/i386/gftp-2.0.8-1.i386.rpm
  ghostscript-5.50-7: ftp://updates.redhat.com/7.0/en/os/i386/ghostscript-5.50-8.i386.rpm
  glibc-2.1.92-14: ftp://updates.redhat.com/7.0/en/os/i386/glibc-2.2-12.i386.rpm
  glibc-devel-2.1.92-14: ftp://updates.redhat.com/7.0/en/os/i386/glibc-devel-2.2-12.i386.rpm
  gnorpm-0.9-27: ftp://updates.redhat.com/7.0/en/os/i386/gnorpm-0.95.1-5.i386.rpm
  gnupg-1.0.2-4: ftp://updates.redhat.com/7.0/en/os/i386/gnupg-1.0.6-1.i386.rpm
  iputils-20000418-6: ftp://updates.redhat.com/7.0/en/os/i386/iputils-20001010-1.i386.rpm
  kernel-2.2.16-22: ftp://updates.redhat.com/7.0/en/os/i386/kernel-2.2.19-7.0.1.i386.rpm (already applied above)
  kernel-pcmcia-cs-2.2.16-22: ftp://updates.redhat.com/7.0/en/os/i386/kernel-pcmcia-cs-2.2.19-7.0.1.i386.rpm
  kernel-utils-2.2.16-22: ftp://updates.redhat.com/7.0/en/os/i386/kernel-utils-2.2.19-7.0.1.i386.rpm
  kernel-source-2.2.16-22: ftp://updates.redhat.com/7.0/en/os/i386/kernel-source-2.2.19-7.0.1.i386.rpm
  krb5-devel-1.2.1-8: ftp://updates.redhat.com/7.0/en/os/i386/krb5-devel-1.2.2-5.i386.rpm
  krb5-libs-1.2.1-8: ftp://updates.redhat.com/7.0/en/os/i386/krb5-libs-1.2.2-5.i386.rpm
  losetup-2.10m-5: ftp://updates.redhat.com/7.0/en/os/i386/losetup-2.10r-5.i386.rpm
  LPRng-3.6.22-5: ftp://updates.redhat.com/7.0/en/os/i386/LPRng-3.7.4-23.i386.rpm
  man-1.5h1-10: ftp://updates.redhat.com/7.0/en/os/i386/man-1.5i-4.i386.rpm
  minicom-1.83.1-4: ftp://updates.redhat.com/7.0/en/os/i386/minicom-1.83.1-8.i386.rpm
  modutils-2.3.14-3: ftp://updates.redhat.com/7.0/en/os/i386/modutils-2.3.21-1.i386.rpm
  mount-2.10m-5: ftp://updates.redhat.com/7.0/en/os/i386/mount-2.10r-5.i386.rpm
  mutt-1.2.5i-3: ftp://updates.redhat.com/7.0/en/os/i386/mutt-1.2.5i-8.7.i386.rpm
  ncurses-5.1-2: ftp://updates.redhat.com/7.0/en/os/i386/ncurses-5.2-2.i386.rpm
  ncurses-devel-5.1-2: ftp://updates.redhat.com/7.0/en/os/i386/ncurses-devel-5.2-2.i386.rpm

netscape-common-4.75-2: ftp://updates.redhat.com/7.0/en/os/i386/netscape-common-4.77-1.i386.rpm
netscape-communicator-4.75-2: ftp://updates.redhat.com/7.0/en/os/i386/netscape-communicator-4.77-1.i386.rpm
nfs-utils-0.1.9.1-7: ftp://updates.redhat.com/7.0/en/os/i386/nfs-utils-0.3.1-7.i386.rpm
openssh-2.1.1p4-1: ftp://updates.redhat.com/7.0/en/os/i386/openssh-2.5.2p2-1.7.2.i386.rpm
openssh-askpass-2.1.1p4-1: ftp://updates.redhat.com/7.0/en/os/i386/openssh-askpass-2.5.2p2-1.7.2.i386.rpm
openssh-askpass-gnome-2.1.1p4-1: ftp://updates.redhat.com/7.0/en/os/i386/openssh-askpass-gnome-2.5.2p2-1.7.2.i386.rpm
openssh-clients-2.1.1p4-1: ftp://updates.redhat.com/7.0/en/os/i386/openssh-clients-2.5.2p2-1.7.2.i386.rpm
openssh-server-2.1.1p4-1: ftp://updates.redhat.com/7.0/en/os/i386/openssh-server-2.5.2p2-1.7.2.i386.rpm
pam-0.72-26: ftp://updates.redhat.com/7.0/en/os/i386/pam-0.72-37.i386.rpm
pine-4.21-23: ftp://updates.redhat.com/7.0/en/os/i386/pine-4.33-7.i386.rpm
popt-1.6-4: ftp://updates.redhat.com/7.0/en/os/i386/popt-1.6.2-7x.i386.rpm
rp-pppoe-2.2-4: ftp://updates.redhat.com/7.0/en/os/i386/rp-pppoe-2.5-1.i386.rpm
rpm-4.0-4: ftp://updates.redhat.com/7.0/en/os/i386/rpm-4.0.2-7x.i386.rpm
rpm-build-4.0-4: ftp://updates.redhat.com/7.0/en/os/i386/rpm-build-4.0.2-7x.i386.rpm
rpm-devel-4.0-4: ftp://updates.redhat.com/7.0/en/os/i386/rpm-devel-4.0.2-7x.i386.rpm
rpm-python-4.0-4: ftp://updates.redhat.com/7.0/en/os/i386/rpm-python-4.0.2-7x.i386.rpm
sgml-tools-1.0.9-8: ftp://updates.redhat.com/7.0/en/os/i386/sgml-tools-1.0.9-9.i386.rpm
slocate-2.2-5: ftp://updates.redhat.com/7.0/en/os/i386/slocate-2.4-1.i386.rpm
slrn-0.9.6.2-9: ftp://updates.redhat.com/7.0/en/os/i386/slrn-0.9.6.4-0.7.i386.rpm
stunnel-3.8-4: ftp://updates.redhat.com/7.0/en/os/i386/stunnel-3.10-2.i386.rpm
tcsh-6.09-6: ftp://updates.redhat.com/7.0/en/os/i386/tcsh-6.10-1.i386.rpm
tmpwatch-2.5.1-3: ftp://updates.redhat.com/7.0/en/os/i386/tmpwatch-2.6.2-1.7.i386.rpm
usermode-1.35-2: ftp://updates.redhat.com/7.0/en/os/i386/usermode-1.37-2.i386.rpm
xinetd-2.1.8.9pre9-6: ftp://updates.redhat.com/7.0/en/os/i386/xinetd-2.3.0-1.71.i386.rpm
vixie-cron-3.0.1-56: ftp://updates.redhat.com/7.0/en/os/i386/vixie-cron-3.0.1-61.i386.rpm

User Accounts

- Shadow passwords with md5 hashing are enabled. To Pass: The file /etc/shadow exists, it is mode 600, and an md5 hash performed on a particular password yields the same hash as the corresponding entry in the /etc/shadow file.

```
audithost# ls -l /etc/shadow
-rw-------   1 root     root         713 Jul  1 18:00 /etc/shadow

-------------------Example below is from work on a FreeBSD system --------------------
$ cat c.c
#define _XOPEN_SOURCE
#include <unistd.h>

int main(int argc, char *argv[])
{
   char * passwd;

   if (argc == 3) {
      passwd = crypt(argv[1], argv[2]);
```

```
        printf("key '%s' salt '%s' pass '%s'\n",
        argv[1], argv[2], passwd);
    }
return (0);
}
```

```
$ cc -o c c.c -lcrypt -lc
$ ./c VerySilly '$1$Q1jK19l6'
key 'VerySilly' salt '$1$Q1jK19l6' pass '$1$Q1jK19l6$95q6YDp1kqWZZ4u7i2yV5/'
-------------------------------------end of FreeBSD example----------------------------------
```

```
audithost# grep mary  /etc/shadow
mary:$1$Q1jK19l6$95q6YDp1kqWZZ4u7i2yV5/:11522::99999::::
```

Analysis: This check **Passed.** Let me explain. The first part of the password field in the shadow file (the bit up until the third $), is the salt, or seed. That, combined with the ascii string that is the actual user password ("VerySilly", in this case) is used to compute the hash value. The crypt(3) manpage on Red Hat 7 says that it uses the DES encryption algorithm (see Appendix B); however I'm convinced that the library function crypt(3) does indeed use MD5 when the salt starts with 1, as it does in the case of this shadow file. Since I could not get hold of the source code for the Red Hat 7 version of crypt(3), I decided to look for another way to prove it. (Being tenacious, I like a good challenge, especially at midnight.) I had access to a FreeBSD system, where the manpage and the library function source code agreed. They each stated that the algorithm used by the library function crypt(3) is MD5 if the password salt starts with 1. (See Appendix C for the text of both the source code and the manpage.) The C program text was formed by studying the manpage; it's a pretty simple program. After compiling, it was run with the ascii password and the salt as the two arguments. The result gives the MD5 hashed password value. Notice it is the same value as the value in /etc/shadow.

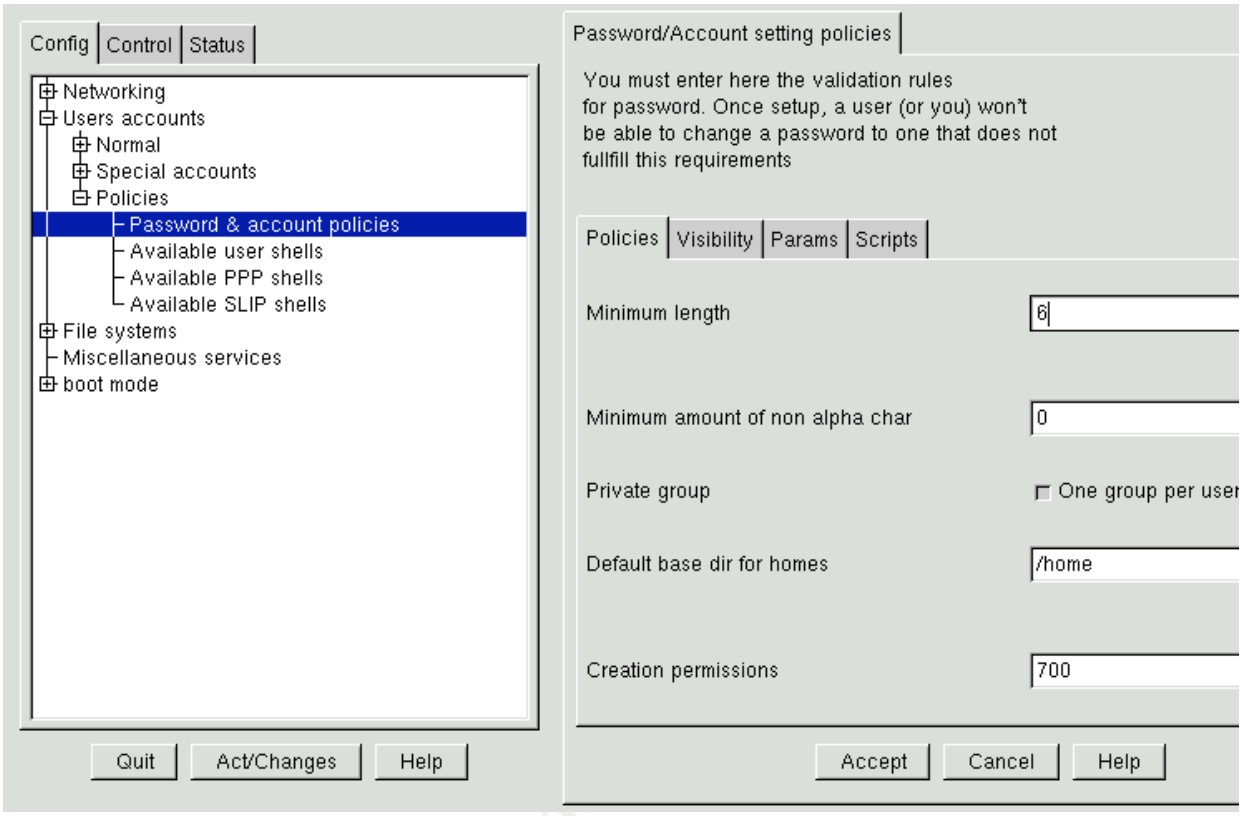
- There is a password for every user and every user's password is shadowed. Use **pwck** to verify passwd and shadow files. To Pass: Yes, **pwck** output shows no errors.

```
# pwck
user adm: directory /var/adm does not exist
user news: directory /var/spool/news does not exist
user uucp: directory /var/spool/uucp does not exist
user gopher: directory /usr/lib/gopher-data does not exist
user gdm: directory /home/gdm does not exist
pwck: no changes
```

Analysis: The output shows that the home directories defined for some users in /etc/passwd do not exist. But we were looking for inconsistencies between the /etc/passwd and the /etc/shadow files, and since we found none, this check **Passed**.

Recommendation: You may want to create the directories the output shows are needed. Particularly, I would create the /var/adm directory. If you are planning to run news, uucp, gopher, etc, it's probably a good idea to create those directories too.

- The system password policy matches the security policy. To Pass: Yes, *linuxconf* reflects password constraints from security policy: Examine User Accounts => Policies => Password & Account Policies screen of *linuxconf*.



Analysis: This check **Failed** because according to the security policy, a user password must be at least 7 characters long with at least one alpha and one non-alpha character.

Recommendation: Use linuxconf to change the policy so that the minimum password length is 7 and the minimum amount of non alpha characters is 1. It is not possible to use linuxconf to enforce the other restriction, at least one alpha, however. (With the linuxconf policy, a user could have a password of 7 digits, for example.) It might be good from time to time to run a password cracking tool to make sure this restriction is being followed.

System Configuration

- There are no shared filesystems. To Pass: Yes, there are no export filesystems configured in **/etc/exports** and the command **exportfs** produces no output.

```
$ ls -l /etc/exp*
-rw-r--r-- 1 root root 0 Jan 12 2000 /etc/exports
$ cat /etc/exports
$ exportfs
$
```

Analysis: This check **Passed**!

- There are no nfs mounted filesystems. To Pass: Yes, the command **df -k** reveals no nfs mounted filesystems.

```
$ df -k
Filesystem 1k-blocks Used Available Use% Mounted on
/dev/hda2  2411408  1021416 1267500 45%  /
```

Analysis: This check **Passed**!

- There are no setuid or setgid files on the system. To Pass: Yes, the commands **find / -perm -4000 -print** and **find / -perm -2000 -print** do not produce any output.

```
# find / -perm -4000 –print
/bin/su
/sbin/pwdb_chkpwd
/sbin/unix_chkpwd
/usr/X11R6/bin/Xwrapper
/usr/bin/chage
/usr/bin/gpasswd
/usr/bin/crontab
/usr/bin/kcheckpass
/usr/bin/suidperl
/usr/bin/sperl5.6.0
/usr/bin/ssh
/usr/bin/passwd
/usr/bin/procmail
/usr/bin/chfn
/usr/bin/chsh
/usr/bin/newgrp
/usr/sbin/sendmail
/usr/sbin/userhelper

# find / -perm -2000 -print
#
```

  Analysis: This check **Failed** because there are several commands with setuid permissions. Recommendation: Configure sudo, by making an /etc/sudoers file of users who are allowed to become root. Once you do this, a user will no longer need to become root using /bin/su; instead the user will invoke sudo when he or she wants to execute a privileged command. Once sudo is configured, you can remove the setuid mode from the files above.

- Is the LILO prompt password protected. To Pass: Yes, the two directives, password and restricted following the prompt directive appear in **/etc/lilo.conf** .

```
$ cat /etc/lilo.conf
boot=/dev/hda
two map=/boot/map
install=/boot/boot.b
prompt
timeout=50
message=/boot/message
linear
default=dos

image=/boot/vmlinuz-2.2.16-22
      label=linux
      read-only
      root=/dev/hda2

other=/dev/hda1
      label=dos
```

  Analysis: This check **Failed** because the required *password* and *restricted* directives do not appear in the file. Recommendation: Add the required directives to the file.

- Are the file permissions of /etc/lilo.conf 600 (rw by owner only) and is the file owned by root? To Pass: Yes, **ls –l /etc/lilo.conf** output shows -rw------ and owned by root.

  ```
  # ls -l /etc/lilo.conf
  -rw-r--r--    1 root     root          207 Jun 29 16:56 /etc/lilo.conf
  ```

  Analysis: This check **Failed** because the file is readable by group and other. Recommendation: As root, execute the command **chmod 600 /etc/lilo.conf**.

- Does /etc/inittab show that reboot from console with Ctrl+Alt+Del is disabled? To Pass: Yes, the directive allowing this (the default ca directive) has been commented out.

  ```
  # Trap CTRL-ALT-DELETE
  ca::ctrlaltdel:/sbin/shutdown -t3 -r now
  ```

  Analysis: This check **Failed** because the required directive has not been commented out. Recommendation: Comment out the *ca* directive.

- Does /etc/inittab show that the root password is required to enter single user mode? To Pass: Yes, the *wait* directive has been added below the *sysinit* directive.

  ```
  # System initialization.
  si::sysinit:/etc/rc.d/rc.sysinit
  ```

  Analysis: This check **Failed** because the required directive has not been added. Recommendation: Add the *wait* directive after the *sysinit* directive.

- Does /etc/security/access.conf show that all console logins except for root and john are disabled? To Pass: The All EXCEPT directive should be configured to include root and john (users who are allowed to login at the console).

  ```
  # Disallow console logins to all but a few accounts.
  #
  #-:ALL EXCEPT wheel shutdown sync:console
  ```

  Analysis: This check **Failed** because the ALL EXCEPT directive was unconfigured (commented out). Recommendation: Configure this directive by uncommenting it and allowing only root and john to login at the console.

- Are TCP wrappers in use to manage access to network services? To Pass: the last line in ***/etc/hosts.allow*** is **deny all** and the only uncommented line in ***/etc/hosts.deny*** is **ALL: ALL**.

  ```
  #
  # hosts.deny  This file describes the names of the hosts which are
  #             *not* allowed to use the local INET services, as decided
  #             by the '/usr/sbin/tcpd' server.
  #
  # The portmap line is redundant, but it is left to remind you that
  # the new secure portmap uses hosts.deny and hosts.allow.  In particular
  # you should know that NFS uses portmap!

  #
  # hosts.allow This file describes the names of the hosts which are
  #             allowed to use the local INET services, as decided
  #             by the '/usr/sbin/tcpd' server.
  #
  ```

Analysis: This check **Failed** because both hosts.deny and hosts.allow are empty (have not been configured). Recommendation: Configure TCP wrappers properly by editing both /etc/hosts.allow and /etc/hosts.deny, adding the proper directives.

- Does /etc/securetty show that remote (telnet) users are not allowed? To Pass: Yes, **/etc/securetty** shows file only contains *tty* entries.

```
$ cat /etc/securetty
vc/1
vc/2
vc/3
vc/4
vc/5
vc/6
vc/7
vc/8
vc/9
vc/10
vc/11
tty1
tty2
tty3
tty4
tty5
tty6
tty7
tty8
tty9
tty10
tty11
```

Analysis: This check **Failed** because vc entries allow remote connections. Recommendation: Edit the file so that only tty entries appear.

Services/Ports

- From the inside: what ports are open? Run **nmap -I -O -sR localhost**. To Pass: port 22 for ssh.

```
$ nmap -I -O -sR localhost

Starting nmap V. 2.53 by fyodor@insecure.org ( www.insecure.org/nmap/ )
Interesting ports on localhost.localdomain (127.0.0.1):
(The 1510 ports scanned but not shown below are in state: closed)
Port       State      Service (RPC)          Owner
21/tcp     open       ftp                    root
22/tcp     open       ssh                    root
23/tcp     open       telnet                 root
25/tcp     open       smtp                   root
79/tcp     open       finger                 root
111/tcp    open       sunrpc (rpcbind V2)    rpc
113/tcp    open       auth                   nobody
513/tcp    open       login                  root
514/tcp    open       shell                  root
515/tcp    open       printer                lp
587/tcp    open       submission             root
1024/tcp   open       kdm (status V1)        rpcuser
```

```
6000/tcp   open       X11                 root

TCP Sequence Prediction: Class=random positive increments
                         Difficulty=2276003 (Good luck!)
Remote operating system guess: Linux 2.1.122 - 2.2.14
Nmap run completed -- 1 IP address (1 host up) scanned in 12 seconds
```

Analysis: This check **Failed** since we see ports open that shouldn't be open. Port 6000, running X11 is troubling. There is a mail server on port 25, and telnet, ftp, finger, printer, auth, rpc/kdm, sunrpc, and submission (that doesn't sound good, does it?), as well as the remote services login and shell all have open ports. Recommendation: Close all ports except port 22. Check the /etc/services file to see the default ports for each service; modify this file if necessary. Ports other than 22 will be closed when the ports' corresponding services are turned off. See below.

- From the inside: what services are running? Run **netstat -at**. To Pass: ssh should be the only service in LISTEN mode.

```
$ netstat -at
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address  Foreign Address        State
tcp       57      0 audithost:4055 vic.cc.purdue.edu:ftp CLOSE_WAIT
tcp        0      0 *:587                *:*                    LISTEN
tcp        0      0 audithost:auth  otherhost:33550     CLOSE
tcp        1      0 audithost:3926  openssh.sunsite.ual:www CLOSE_WAIT
tcp        1      0 audithost:3925  openssh.sunsite.ual:www CLOSE_WAIT
tcp        1      0 audithost:3924  openssh.sunsite.ual:www CLOSE_WAIT
tcp        0      0 audithost:auth  otherhost:51101     CLOSE
tcp        0      0 *:X                  *:*                    LISTEN
tcp        0      0 audithost:telnet otherhost:32867     ESTABLISHED
tcp        0      0 *:smtp               *:*                    LISTEN
tcp        0      0 *:printer            *:*                    LISTEN
tcp        0      0 *:ssh                *:*                    LISTEN
tcp        0      0 *:login              *:*                    LISTEN
tcp        0      0 *:shell              *:*                    LISTEN
tcp        0      0 *:telnet             *:*                    LISTEN
tcp        0      0 *:ftp                *:*                    LISTEN
tcp        0      0 *:finger             *:*                    LISTEN
tcp        0      0 *:auth               *:*                    LISTEN
tcp        0      0 *:1024               *:*                    LISTEN
tcp        0      0 *:sunrpc             *:*                    LISTEN
```

Analysis: This check **Failed** since we see unneeded services are running. Here is a list of services that shouldn't be running:

- smtp – This is a mail server. We can send and receive email without running a server.
- printer – This is a print server. We can send jobs to a remote printer without have this service on.
- login – This is actually rlogin. Bad.
- shell – This is rsh. Bad.
- telnet – "telnet is really really bad."[1]
- ftp – This is a workstation and probably doesn't need this service running.

[1] The Bastille user interface. Lasser, John and Beale, Jay. "Bastille Linux Hardening System 1.2.0". URL: http://bastille-linux.sourceforge.net/

- finger – This is a bad thing to have on. Remote persons can see who is logged in and usually what type of system it is.
- sunrpc – We don't need this, it's the port mapper. May as well turn it off.
- and a few others above we'll try to get to the bottom of: submission, 1024, auth, 587, X.

Recommendation: Compare this list with the nmap output to give more clues about what these services are. We want to disable everything except for ssh. See below.

- List all open network sockets. Are there any open sockets that you should be concerned about? Use **lsof –i +M**. To Pass: ssh and browser files, and some others may also be acceptable. Output must be carefully examined.

```
$ lsof -i +M
COMMAND    PID USER   FD   TYPE DEVICE SIZE NODE NAME
portmap    348 root    3u  IPv4    495       UDP *:sunrpc[portmapper]
portmap    348 root    4u  IPv4    496       TCP *:sunrpc[portmapper] (LISTEN)
rpc.statd  375 root    4u  IPv4    527       UDP *:975
rpc.statd  375 root    6u  IPv4    533       UDP *:1025[status]
rpc.statd  375 root    7u  IPv4    536       TCP *:1024[status] (LISTEN)
identd     444 root    4u  IPv4    590       TCP *:auth (LISTEN)
identd     444 root   11u  IPv4   8735       TCP audithost:auth->otherhost:51101 (CLOSE)
identd     444 root   12u  IPv4  10314       TCP audithost:auth->otherhost:33550 (CLOSE)
identd     446 root    4u  IPv4    590       TCP *:auth (LISTEN)
identd     446 root   11u  IPv4   8735       TCP audithost:auth->otherhost:51101 (CLOSE)
identd     446 root   12u  IPv4  10314       TCP audithost:auth->otherhost:33550 (CLOSE)
identd     448 root    4u  IPv4    590       TCP *:auth (LISTEN)
identd     448 root   11u  IPv4   8735       TCP audithost:auth->otherhost:51101 (CLOSE)
identd     448 root   12u  IPv4  10314       TCP audithost:auth->otherhost:33550 (CLOSE)
identd     450 root    4u  IPv4    590       TCP *:auth (LISTEN)
identd     450 root   11u  IPv4   8735       TCP audithost:auth->otherhost:51101 (CLOSE)
identd     450 root   12u  IPv4  10314       TCP audithost:auth->otherhost:33550 (CLOSE)
identd     451 root    4u  IPv4    590       TCP *:auth (LISTEN)
identd     451 root   11u  IPv4   8735       TCP audithost:auth->otherhost:51101 (CLOSE)
identd     451 root   12u  IPv4  10314       TCP audithost:auth->otherhost:33550 (CLOSE)
xinetd     506 root    3u  IPv4    658       TCP *:finger (LISTEN)
xinetd     506 root    4u  IPv4    659       TCP *:ftp (LISTEN)
xinetd     506 root    5u  IPv4    660       TCP *:telnet (LISTEN)
xinetd     506 root    7u  IPv4    661       TCP *:shell (LISTEN)
xinetd     506 root    8u  IPv4    662       TCP *:login (LISTEN)
sshd       537 root    3u  IPv4    686       TCP *:ssh (LISTEN)
lpd        558 root    5u  IPv4    709       TCP *:printer (LISTEN)
sendmail   606 root    4u  IPv4    765       TCP *:smtp (LISTEN)
sendmail   606 root    5u  IPv4  10607       TCP *:587 (LISTEN)
in.telnet 1005 root    0u  IPv4   1041       TCP audithost:telnet->otherhost:32867 (ESTABLISHED)
in.telnet 1005 root    1u  IPv4   1041       TCP audithost:telnet->otherhost:32867 (ESTABLISHED)
in.telnet 1005 root    2u  IPv4   1041       TCP audithost:telnet->otherhost:32867 (ESTABLISHED)
X         2198 root    1u  IPv4   8909       TCP *:X (LISTEN)
netscape- 2549  john   23u  IPv4  11926       TCP audithost:4055->vic.cc.purdue.edu:ftp (CLOSE_WAIT)
```

Analysis: This check **Failed** because there are lots of things here we don't want to see! Here we solve a few of the mysteries that appeared in the netstat output. We can see which daemons are controlling which ports and services!

- We can see that some ports which are open that do not have services tied to them are under the control of the rpc.statd daemon: 975/UDP, 1025/UDP, 1024/TCP.

- There is a portmapper running with both TCP and UDP protocols: sunrpc
- The ident daemon controls the authentication service.
- sendmail is running smtp on port 25 and controlling port 587 (this showed up on the nmap output as a service called *submission*).
- The printer server is controlled by lpd.
- Lastly, there's the live telnet session (yikes!), the netscape session, and Mr. X Windows.

Recommendation: Execute the following commands, as root, to stop certain daemons, then rename the corresponding startup scripts in /etc/rc3.d, /etc/rc4.d and /etc/rc5.d:

- /etc/init.d/identd stop and rename S*identd to disabled.S*identd
- /etc/init.d/portmap stop and rename S*portmap to disabled.S*portmap
- /etc/init.d/lpd stop and rename S*lpd to disabled.S*lpd (in /etc/rc2.d also)
- To stop the X service listening on port 6000, follow this procedure:
  1. Edit the /etc/X11/gdm/gdm.conf file and change the servers directive as follows:
     [servers]
     0=/usr/bin/X11/X -nolisten tcp
     #1=/usr/bin/X11/X
  2. Kill the gdm (Gnome Display Manager) process.
  3. If X Windows is running locally on the system, exit it, or Kill X.
  4. Login to the console via X. The new gdm/X configuration will now be active.
- See below for how to unconfigure and shutdown telnet.
- See below for how to reconfigure sendmail so it doesn't run as a daemon.

- From the outside: do a port scan. Which ports are reported open? Use **nessus**. To Pass: only port 22 is visible. Analysis: Nessus Example 1 shows that this check **Failed** because many ports and corresponding services are visible. Recommendation: Close all listening ports except 22 (ssh). This will be taken care of when we stop unnecessary services. See below.

- From the outside: do a vulnerability assessment using nessus. Which services are vulnerable? See Nessus Examples 1 and 2 for screen shots. To Pass: the nessus output reveals no vulnerabilities; the ssh version is up to date as far as security releases are concerned. This check **Failed** because the version of openssh is suspect, nessus reporting that the version of openssh is lower than 2.3.0 (nessus' recommended version of openssh). The vulnerability cited is CAN-2001-0144. The CVE entry for this can be found at URL http: //cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2001-0144. I compared the nessus output to the system's self reporting of its ssh version:

```
$ rpm -qa |grep openssh
openssh-2.1.1p4-1
openssh-server-2.1.1p4-1
openssh-askpass-2.1.1p4-1
openssh-clients-2.1.1p4-1
openssh-askpass-gnome-2.1.1p4-1
$ ssh -V
SSH Version OpenSSH_2.1.1, protocol versions 1.5/2.0.
Compiled with SSL (0x0090581f).
```

Analysis: This check **Failed**, since we don't see ssh 1.2.32 or later, or openssh-2.5.2p2-1.7.2 or later. Recommendation: Install the openssh-2.5.2p2-1.7.2. See Appendix A for the full list of security updates.

- Secure remote shell, sshd is running. To Pass: **ps –ef | grep sshd** succeeds (produces output other than the grep process).

```
$ ps -ef | grep sshd
root      501     1  0 Jul12 ?        00:00:05 [sshd]
root    19344   501  0 19:13 ?        00:00:00 [sshd]
root    19403   501  0 19:27 ?        00:00:00 [sshd]
root    19614   501  0 19:37 ?        00:00:00 /usr/sbin/sshd
mary    19728 19653  3 19:54 pts/3    00:00:00 grep sshd
```

  Analysis: This check **Passed**!

- What is the version of ssh? To Pass: **ssh –V** or **rpm –qa | grep ssh** shows ssh 1.2.32 or later, or openssh-2.5.2p2-1.7.2 or later. Analysis: This check **Failed** because the openssh version on the system was 2.1.1p4-1. (Results from this check are printed in the item above.) Recommendation: update the version of openssh – see security updates check above.
- Are unneeded services unconfigured? Use **chkconfig --list** to show, for each run level, which services are configured and whether they are *on* or *off*. To Pass: this output must be examined closely to determine that only necessary services are, and that of those, they are enabled (on) only at the correct run levels (0-6).

```
$ chkconfig --list
anacron 0:off  1:off  2:on   3:on   4:on   5:on   6:off
apmd    0:off  1:off  2:on   3:on   4:on   5:on   6:off
syslog  0:off  1:off  2:on   3:on   4:on   5:on   6:off
crond   0:off  1:off  2:on   3:on   4:on   5:on   6:off
netfs   0:off  1:off  2:off  3:on   4:on   5:on   6:off
network 0:off  1:off  2:on   3:on   4:on   5:on   6:off
random  0:off  1:off  2:on   3:on   4:on   5:on   6:off
rawdevices 0:off 1:off 2:off  3:on   4:on   5:on   6:off
arpwatch  0:off  1:off  2:off  3:off  4:off  5:off  6:off
atd     0:off  1:off  2:off  3:on   4:on   5:on   6:off
xfs     0:off  1:off  2:on   3:on   4:on   5:on   6:off
keytable  0:off 1:off  2:on   3:on   4:on   5:on   6:off
gpm     0:off  1:off  2:on   3:on   4:on   5:on   6:off
ipchains  0:off 1:off  2:on   3:on   4:on   5:on   6:off
irda    0:off  1:off  2:off  3:off  4:off  5:off  6:off
isdn    0:off  1:off  2:on   3:on   4:on   5:on   6:off
pcmcia  0:off  1:off  2:on   3:on   4:on   5:on   6:off
kdcrotate 0:off 1:off  2:off  3:off  4:off  5:off  6:off
kudzu   0:off  1:off  2:off  3:on   4:on   5:on   6:off
linuxconf 0:off 1:off  2:on   3:on   4:on   5:on   6:off
lpd     0:off  1:off  2:on   3:on   4:on   5:on   6:off
nfs     0:off  1:off  2:off  3:off  4:off  5:off  6:off
nfslock  0:off  1:off  2:off  3:on   4:on   5:on   6:off
sshd    0:off  1:off  2:on   3:on   4:on   5:on   6:off
identd  0:off  1:off  2:off  3:on   4:on   5:on   6:off
portmap 0:off  1:off  2:off  3:on   4:on   5:on   6:off
pppoe   0:off  1:off  2:off  3:off  4:off  5:off  6:off
rstatd  0:off  1:off  2:off  3:off  4:off  5:off  6:off
rusersd 0:off  1:off  2:off  3:off  4:off  5:off  6:off
rwalld  0:off  1:off  2:off  3:off  4:off  5:off  6:off
rwhod   0:off  1:off  2:off  3:off  4:off  5:off  6:off
```

```
sendmail 0:off  1:off   2:on    3:on    4:on    5:on    6:off
rhnsd   0:off   1:off   2:off   3:on    4:on    5:on    6:off
xinetd  0:off   1:off   2:off   3:on    4:on    5:on    6:off
ypbind  0:off   1:off   2:off   3:off   4:off   5:off   6:off
yppasswdd 0:off 1:off   2:off   3:off   4:off   5:off   6:off
ypserv  0:off   1:off   2:off   3:off   4:off   5:off   6:off
xinetd based services:
        finger: on
        linuxconf-web:  off
        rexec:  off
        rlogin: on
        rsh:    on
        ntalk:  off
         talk:   off
        telnet: on
        tftp:   off
        wu-ftpd:        on
```

Analysis: Whether a given service is on or off at a given run level corresponds to the presence of a corresponding S* (start) or K* (kill) script in the /etc/rcN.d directories, where N is the run level. Here's a quick summary of what each run level means:
0 : the system is halted
1: the system is in single-user mode
2: the system is in multi-user mode
3: the system is in multi-user mode with network services
4: pretty much the same as above but is left for user definition
5: the system adds X Windows to run level 3
6: the system is in reboot mode

This check **Failed**, since we see many unneeded xinetd services which are configured. But what is all the rest of this stuff? The descriptive text of some of the items below was taken directly from The Red Hat Linux 7 Bible. (See footnote below.)

- anacron – Runs **at** and **batch** jobs that were not run because the computer was down.[1]
- apmd – This run-level daemon controls the Advanced Power Management daemon, which monitors battery status, and which can safely suspend or shut down all or part of a machine that supports it.[1]
- syslog – Starts or stops the klogd and syslogd daemons that handle logging events from the kernel and other processes, respectively.[1]
- crond – Starts or stops the cron daemon to periodically run routine commands.[1]
- netfs – Mounts or unmounts network file systems.[1]
- network – Starts or stops all configured network interfaces and initialized the TCP/IP and IPX protocols.[1]
- random – Loads or saves the current state of the machine's random number generator's random seed to ensure more random randomness.[1]
- rawdevices – Binds raw devices to corresponding block devices.
- arpwatch – Keeps track of ethernet/IP address pairings.

[1] Negus, Christopher. Red Hat Linux 7 Bible. Foster City: IDG Books Worldwide, Inc., 2001, pp. 415-417.

- atd – Starts or stops the **at** daemon to receive, queue, and run jobs submitted via the **at** or **batch** commands.[1]
- xfs – Starts or stops xfs, the X Window font server daemon.[1]
- keytable – Loads the predefined keyboard map.[1]
- gpm – Controls the gpm daemon, which allows the mouse to interact with console- and text-based applications.[1]
- ipchains – This daemon is part of the Linux kernel. It examines packets as they are sent and received on a network interface and decides which packets should be delivered and which should be stopped.[2]
- irda - Starts or stops the irmanager.
- isdn - Starts or stops the daemon which controls the isdn connection/configuration.
- pcmcia - Loads or unloads modules, drivers, and programs to support PCMCIA cards in laptop computers.[1]
- kdcrotate – Rotates the list of KDCs in /etc/krb5.conf.
- kudzu - Detects and configures new hardware at boot time.[1]
- linuxconf - Loads any customized linuxconf policies or modules, as well as a few default modules (for use with the linuxconf system configuration tool).[1]
- lpd – Controls the lpd line printer daemon that handles spooling printing requests.[1]
- nfs – Starts or stops the NFS-related daemons (rpc.nfsd, rpc.mountd, rpc.statd, and rpc.rquotad).[1]
- nfslock – Starts or stops the NFS file locking service.
- sshd – This is the secure shell daemon.
- identd – This daemon looks up specific TCP/IP connections and returns the user name of the process owning the connection.
- portmap – Starts or stops the portmap daemon, which manages programs and protocols that utilize the Remote Procedure Call (RPC) mechanism.[1]
- pppoe – This daemon allows PPP connections over an ethernet connection.

The RPC (Remote Procedure Call) services:

- rstatd – Starts or stops the rpc.rstatd daemon, which enables others on the network to probe the machine for performance statistics.[1]
- rusersd – Starts or stops the rpc.rusersd daemon, which enables others on the network to locate users on the machine.[1]
- rwalld – Starts or stops the daemon which is invoked in order to write messages to currently logged-in users.
- rwhod - Starts or stops the rwhod daemon, which enables others on the network to obtain a list of all currently logged-in users.[1]

Sendmail, network services:

- sendmail – Controls the sendmail daemon, which handles incoming and outgoing SMTP mail messages.[1]

---

[1] Negus, Christopher. Red Hat Linux 7 Bible. Foster City: IDG Books Worldwide, Inc., 2001, pp. 415-417.
[2] Negus, Christopher. Red Hat Linux 7 Bible. Foster City: IDG Books Worldwide, Inc., 2001, p. 492.

- rhnsd - Starts or stops the Red Hat Network Services daemon. This handles connecting periodically to the RHNS servers to check for updates.
- xinetd – Sets the machine's hostname, establishes network routes, and controls xinetd, the network services daemon which listens for incoming TCP/IP connections to the machine.[1]

And, yp (yellow pages), aka NIS (Network Information Service):

- ypbind – Binds to an NIS master server (if NIS is configured), and starts or stops the ypbind process, which communicates with the master server.[1]
- yppasswdd – (Used by the NIS server) This script starts the rpc.yppasswdd daemon. This daemon handles requests from users on NIS client computers who want to change their user passwords.[3]
- ypserv – (Used by the NIS server) This script starts the ypserv daemon. It reads information from the /etc/ypserv.conf file to determine what to do. Then it listens for requests from NIS client computers on the network.[3]

Recommendation: In addition to the recommendation from the lsof check above, the xinetd services have to go! See below.

- Have unneeded services been shut down properly?

DNS: This service has been stopped, unconfigured and remove from the system.

- There is no DNS server running. To Pass: Yes, **ps –ef | grep named** shows no output.

```
$ ps –ef | grep named | grep -v grep
$
```

Analysis: This check **Passed**!

- The naming service software been removed from the box. To Pass: Yes, rpm -qa does not show *bind* or *caching-nameserver* packages.

  Analysis: This check **Passed**!

- Remote nameservers have been configured. To Pass: Yes, /etc/resolv.conf defines the IP addresses of primary and secondary nameservers.

  Analysis: This check **Passed**!

FTP: The ftp service is not running, is unconfigured and the software has been uninstalled (removed).

- There is no FTP service running. . To Pass: **netstat -at | grep ftp** does not produce output.

```
# netstat -at | grep ftp
tcp        0      0 *:ftp                   *:*                     LISTEN
```

  Analysis: This check **Failed** because the ftp service is running.

  Recommendation: Shutdown the ftp service. See below.

- Does /etc/ftpusers show that root and system accounts cannot use ftp? ? To Pass: contents of /etc/ftpusers should include root and any privileged users.

[1] Negus, Christopher. Red Hat Linux 7 Bible. Foster City: IDG Books Worldwide, Inc., 2001, pp. 415-417.
[3] Negus, Christopher. Red Hat Linux 7 Bible. Foster City: IDG Books Worldwide, Inc., 2001, p.795.

```
$ cat /etc/ftpusers
root
bin
daemon
adm
lp
sync
shutdown
halt
mail
news
uucp
operator
games
nobody
```

Analysis: This check **Passed**!

- The file /etc/ftpaccess does not allow guest or anonymous connections (even though you have turned off the ftp service). To Pass: Specific user types should not appear in the *class* directive.

```
$ cat /etc/ftpaccess
class   all   real,guest,anonymous  *

email root@localhost

loginfails 5

readme  README*    login
readme  README*    cwd=*

message /welcome.msg           login
message .message               cwd=*

compress        yes             all
tar             yes             all
chmod           no              guest,anonymous
delete          no              guest,anonymous
overwrite       no              guest,anonymous
rename          no              guest,anonymous

log transfers anonymous,real inbound,outbound

shutdown /etc/shutmsg

passwd-check rfc822 warn
```

Analysis: This check **Failed** because user types *real, guest* and *anonymous* appear in the *class* directive. Recommendation: Remove user types from the class directive.

- The following configuration files have been removed from the /etc/inetd.d directory: tftp, wu-ftpd. To Pass**: ls /etc/xinetd.d/*ftp*** produces no output.

```
$ ls /etc/inetd.d/*ftp*
tftp
wu-ftpd
```

Analysis: This check **Failed** because these configuration files existed. Recommendation: Delete these configuration files.

- The ftp packages have been removed from the system. To Pass: **rpm –qa | grep ftp** does not produce output.

```
$ rpm -qa | grep ftp
gftp-2.0.7b-2
ncftp-3.0.1-7
ftp-0.17-6
tftp-server-0.17-5
wu-ftpd-2.6.1-6
anonftp-3.0-9
```

Analysis: This check **Failed** because there are several packages installed relating to ftp. (The tftp-server and the anonftp are the most serious problems.) Recommendation: Remove these packages from the system using the following command iteratively, starting with anonftp and then by replacing anonftp with the other simple package names:

```
# rpm –e anonftp
```

- TELNET, FINGER, TALK, RLOGIN, RSH, REXEC: These services are not running, are unconfigured and the software has been uninstalled (removed). For each of these services, check the following (substitute the name of the service):

  - These services are not running. To Pass: Yes, **netstat -at | grep *service*** does not produce output.

```
tcp    0    0 *:login          *:*          LISTEN
tcp    0    0 *:shell          *:*          LISTEN
tcp    0    0 *:telnet         *:*          LISTEN
tcp    0    0 *:finger         *:*          LISTEN
```

Analysis: This check **Failed** because rlogin, rsh, telnet, and finger are running. Recommendation: Unconfigure these services by removing their config files from the xinetd.d directory, then restart xinetd. See below.

  - The following configuration files have been removed from the /etc/inetd.d directory: finger, ntalk, rexec, rlogin, rsh, talk, telnet. To Pass: Yes, **ls /etc/xinetd.d** does not show any of these configuration files.

```
$ ls /etc/xinetd.d
finger
linuxconf-web
ntalk
rexec
rlogin
rsh
talk
telnet
tftp
wu-ftpd
```

Analysis: This check **Failed** because these configuration files are intact. Recommendation: Delete the configuration files corresponding to these services.

- The packages corresponding to these services have been removed from the system. To Pass: Yes, **rpm –qa | grep *service*** does not produce output.

  finger-server-0.17-4
  finger-0.17-4
  talk-server-0.17-7
  talk-0.17-7
  telnet-server-0.17-7
  telnet-0.17-7
  anonftp-3.0-9
  wu-ftpd-2.6.1-6
  tftp-server-0.17-5
  ftp-0.17-6

  Analysis: This check **Failed.** Recommendation: Execute the following as root (where *package* is the name of each package found in the rpm query):

  # rpm –e *package*

- SENDMAIL
  - There is no email server or relay daemon running. To Pass: Yes, In /etc/sysconfig/sendmail, the value for DAEMON is no.
    $ cat /etc/sysconfig/sendmail

    DAEMON=yes
    QUEUE=1h

    Analysis: This check **Failed**. Recommendation: Change the value of DAEMON to no.

  - SMTP vrfy and expn commands cannot be run remotely. To Pass: Yes, /etc/sendmail.cf contains the line **PrivacyOptions=goaway**.
    $ grep Privacy /etc/sendmail.cf
    O PrivacyOptions=authwarnings

    Analysis: This check **Failed**. Recommendation: Change the value of PrivacyOptions to *goaway*.

- OTHER
  - NFS is not running and the NFS packages are not installed on the system. To Pass: Yes**, ps –ef | grep nfs** shows no daemon running, and **rpm -qa | grep nfs** does not produce output.

    $ ps -ef | grep nfs
    root 19800 19653 0 20:07 pts/3 00:00:00 grep nfs

    $ rpm -qa | grep nfs
    nfs-utils-0.1.9.1-7

    Analysis: This check **Failed** because though the daemon isn't running, an nfs package still exists on the system.
    Recommendation: Remove the nfs-utils package by executing, as root, the command **rpm –e nfs-utils**.

  - No print server is running. To Pass: Yes, **ps –ef | grep lpd** produces no output (other than the grep process) and **/etc/hosts.lpd**, if it exists, is blank. (Note, this system should configure its /etc/printcap via printtool, to point to a remote print server; it does not need lpd in order to use a remote printer.)

```
$ ps -ef | grep lpd
lp       522     1  0 Jul12 ?        00:00:00 [lpd]
root     19885 19653  0 20:09 pts/3    00:00:00 grep lpd

$ cat /etc/hosts.lpd
cat: /etc/hosts.lpd: No such file or directory
```

Analysis: This check **Failed** because a print server was running.
Recommendation: As root, execute /etc/init.d/lpd stop. Then, rename the rc*.d scripts that point to /etc/init.d/lpd by prepending the string "disabled". This will prevent the lpd daemon from starting during boot. (It's also a good way to see which startup scripts you've disabled.)

- There is no web server running and the web server packages do not exist on the box. To Pass: Yes, **ps –ef | grep httpd** and **rpm -qa | grep apache** do not produce output.

```
$ ps -ef | grep http
root     19900 19653  0 20:10 pts/3    00:00:00 grep http
$ rpm -qa | grep apache
$
```

Analysis: This check **Passed**!

- If there are no xinetd services running, xinetd is turned off. To Pass: If no xinetd services are running, **ps –ef | grep xinetd** produces no output.

```
# ps –ef | grep xinetd
#
```

Analysis: This check was run after all xinetd services had been unconfigured and xinetd had been shut down by executing the following as root: /etc/init.d/xinetd stop. This check **Passed**!

File Integrity

- There are no **.rhosts** files on the system, in the root file system, or in any user partitions. To Pass: **find / -name .rhosts -print** does not find anything.

```
# find / -name .rhosts -print
#
```

Analysis: This check **Passed**!

- There is no **/etc/hosts.equiv** file. To Pass: **ls –l /etc/hosts.equiv** fails.

```
$ ls -l /etc/hosts.equiv
ls: /etc/hosts.equiv: No such file or directory
```

Analysis: This check **Passed**!

- Is Tripwire installed and configured? To Pass: Yes, Tripwire has been installed and configured.
Analysis: This check **Failed** because Tripwire was not installed. Recommendation: Install and configure Tripwire. The Tripwire policy configuration file is called twpol.txt. The directory list should include any directory which holds immutable files. A good example for starting out can be found in David Hughes' article "Have I Been Hacked?" in Sys Admin Magazine. See References for complete information. (I would add the /etc/xinetd.d directory

to this policy, so that we would know if any of these services we've unconfigured mysteriously become configured again.) The tripwire database should be kept on a removable medium. You could configure a cron job to run a script which would mount a cdrom, run **tripwire --check –d /dev/cdrom**, send the report to the sys admin, then unmount the cdrom. A write-once cdrom would be ideal for this purpose, because the database would not be able to be changed. When it is necessary to update the database, via the **tripwire --update** command, a new fresh cdrom could be used.

Post Recommendations Results

After turning off all the services required to comply with the security policy, here are the results of rerunning a few key checks (also please see Nessus Example 3):

```
# nmap -I -O -sR localhost
Starting nmap V. 2.53 by fyodor@insecure.org ( www.insecure.org/nmap/ )
Interesting ports on localhost.localdomain (127.0.0.1):
(The 1522 ports scanned but not shown below are in state: closed)
Port       State       Service (RPC)          Owner
22/tcp     open        ssh

TCP Sequence Prediction: Class=random positive increments
                         Difficulty=2427860 (Good luck!)
Remote operating system guess: Linux 2.1.122 - 2.2.14
Nmap run completed -- 1 IP address (1 host up) scanned in 1 second

# netstat -at
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0     20 audithost:ssh   otherhost:1597      ESTABLISHED
tcp        0      0 *:ssh                 *:*                     LISTEN

# lsof -i +M
COMMAND  PID USER   FD   TYPE DEVICE SIZE NODE NAME
sshd     500 root    3u  IPv4    557       TCP *:ssh (LISTEN)
sshd    9868 root    4u  IPv4  11938       TCP audithost:ssh->otherhost:1597 (ESTABLISHED)

# chkconfig --list
anacron         0:off   1:off   2:on    3:on    4:on    5:on    6:off
apmd            0:off   1:off   2:on    3:on    4:on    5:on    6:off
syslog          0:off   1:off   2:on    3:on    4:on    5:on    6:off
crond           0:off   1:off   2:on    3:on    4:on    5:on    6:off
netfs           0:off   1:off   2:off   3:on    4:on    5:on    6:off
network         0:off   1:off   2:on    3:on    4:on    5:on    6:off
random          0:off   1:off   2:on    3:on    4:on    5:on    6:off
rawdevices      0:off   1:off   2:off   3:on    4:on    5:on    6:off
arpwatch        0:off   1:off   2:off   3:off   4:off   5:off   6:off
atd             0:off   1:off   2:off   3:on    4:on    5:on    6:off
xfs             0:off   1:off   2:on    3:on    4:on    5:on    6:off
keytable        0:off   1:off   2:on    3:on    4:on    5:on    6:off
gpm             0:off   1:off   2:on    3:on    4:on    5:on    6:off
ipchains        0:off   1:off   2:on    3:on    4:on    5:on    6:off
irda            0:off   1:off   2:off   3:off   4:off   5:off   6:off
isdn            0:off   1:off   2:on    3:on    4:on    5:on    6:off
pcmcia          0:off   1:off   2:on    3:on    4:on    5:on    6:off
kdcrotate       0:off   1:off   2:off   3:off   4:off   5:off   6:off
```

```
kudzu          0:off   1:off   2:off   3:on    4:on    5:on    6:off
linuxconf      0:off   1:off   2:on    3:on    4:on    5:on    6:off
lpd            0:off   1:off   2:off   3:off   4:off   5:off   6:off
nfs            0:off   1:off   2:off   3:off   4:off   5:off   6:off
nfslock        0:off   1:off   2:off   3:on    4:on    5:on    6:off
sshd           0:off   1:off   2:on    3:on    4:on    5:on    6:off
identd         0:off   1:off   2:off   3:on    4:on    5:on    6:off
portmap        0:off   1:off   2:off   3:on    4:on    5:on    6:off
pppoe          0:off   1:off   2:off   3:off   4:off   5:off   6:off
rstatd         0:off   1:off   2:off   3:off   4:off   5:off   6:off
rusersd        0:off   1:off   2:off   3:off   4:off   5:off   6:off
rwalld         0:off   1:off   2:off   3:off   4:off   5:off   6:off
rwhod          0:off   1:off   2:off   3:off   4:off   5:off   6:off
sendmail       0:off   1:off   2:off   3:off   4:off   5:off   6:off
rhnsd          0:off   1:off   2:off   3:on    4:on    5:on    6:off
xinetd         0:off   1:off   2:off   3:on    4:on    5:on    6:off
ypbind         0:off   1:off   2:off   3:off   4:off   5:off   6:off
yppasswdd      0:off   1:off   2:off   3:off   4:off   5:off   6:off
ypserv         0:off   1:off   2:off   3:off   4:off   5:off   6:off
xinetd based services:
#
```

Subjective items (a subset)

- There are no weak passwords on the system. Use a password cracker. To Pass: The password cracker reports no weak passwords.

  Analysis: This check **Failed** because the password belonging to user *mary* was weak. The password matched common dictionary entries. Recommendation: In addition to setting the password policy via linuxconf, users should be trained to know how to make a good and easy-to-remember password. The rule of thumb I use is to tell users to think of a phrase or line from a song they can remember. Then, choose the first letter of each word, and include punctuation, to form the password. For example, "Do you feel lucky, punk?" becomes "Dyfl,p?" - not easy to guess but easy to remember.

- Is an adequate job being done of writing to logs? To Pass: Examine /etc/syslog.conf. List files in /var/log. Questions to answer: What is being logged? How often is it logged? How long are logs kept? Are logs backed up? Are you comfortable with all the answers?

```
# cat /etc/syslog.conf
# Log all kernel messages to the console.
# Logging much else clutters up the screen.
#kern.*                                 /dev/console
# Log anything (except mail) of level info or higher.
# Don't log private authentication messages!
*.info;mail.none;authpriv.none          /var/log/messages
# The authpriv file has restricted access.
authpriv.*                              /var/log/secure
# Log all the mail messages in one place.
mail.*                                  /var/log/maillog
# Log cron stuff
cron.*                                  /var/log/cron
# Everybody gets emergency messages, plus log them on another
# machine.
*.emerg                                 *
```

```
# Save mail and news errors of level err and higher in a
# special file.
uucp,news.crit                          /var/log/spooler
# Save boot messages also to boot.log
local7.*                                /var/log/boot.log

# ls –lt /var/log
-rw-------   1 root     root        1994 Jul 22 14:01 cron
-rw-------   1 root     root        4727 Jul 22 14:01 messages
-rw-------   1 root     root          55 Jul 22 13:42 boot.log
-rw-r--r--   1 root     root      146292 Jul 22 13:41 lastlog
-rw-rw-r--   1 root     utmp      206976 Jul 22 13:41 wtmp
-rw-------   1 root     root          0 Jul 22 10:52 secure
-rw-------   1 root     root      63380 Jul 22 10:52 cron.1
-rw-------   1 root     root     811705 Jul 22 10:52 messages.1
-rw-------   1 root     root      32668 Jul 22 10:47 boot.log.1
-rw-r--r--   1 root     root       2974 Jul 22 10:47 dmesg
-rw-------   1 root     root       1920 Jul 19 20:46 secure.1
```

Analysis: This check **Passed** because an adequate job is being done to write to logs. Here's a summary of what each log file contains:

- cron: status messages from the daemon that runs scheduled jobs via crontab files
- messages: status messages from many different system programs
- boot.log: status messages showing when the system was booted.
- lastlog: login information including date and time for each user.
- wtmp: login information about who is currently logged in
- dmesg: status messages from the kernel during boot
- secure: login information including date of time of attempted logins and sessions

New cron, messages, secure, and boot log files are started when the system boots, and old log files are renamed by appending .N, where N is the number of reboots since that log file was last written to. The logs lastlog, and wtmp continue to grow. The dmesg log is a subset of messages. All these logs are ascii format except for *lastlog*, which is read by executing the **last** command, and *wtmp*, which is read by executing the **who** command with **/var/log/wtmp** as an argument.

Recommendation: For this workstation, I would archive the logs monthly. If you want to write logs to a syslog server, you would replace the filenames in /etc/syslog.conf with @loghost, and define loghost in the audithost's /etc/hosts file.

- Is an adequate job being done to monitor logs? To Pass: Examine swatch configuration and output. Questions to answer: Is swatch configured and running? Which logs is it monitoring? What actions, if any, are taken?
  Analysis: This check **Failed** because I could not get swatch to install and run properly.
  Recommendation: Use a different method to monitor logs. For example, write your own Perl scripts to process and maintain log files. I would pay most attention to the logs relating to tracking users: secure, wtmp, lastlog, and run scripts to process logs in a cron job daily. It would be feasible to have the monitoring script send a short email report to the system administrator of the system.

- Is an adequate job being done regarding backups? To Pass: What is being backed up? How often? Where are backups kept? Has the restore procedure been tested? Are you comfortable with the answers to these questions?

  Analysis: This check **Failed** because no files are currently being backed up.
  Recommendation: I would do weekly backups of key areas of the system. I would select system configuration files (things in /etc), and perhaps executables in the various bin directories.

Evaluating the Audit

I believe that my audit evaluated the security of the system I chose. Since it's a workstation in a company's intranet, I feel my checklist and procedures adequately covered the basics.

One area which was addressed in detail was coming up with a list of security updates which needed to be applied to the system. I feel the list clearly showed where to get the updates and how to apply them. Another area which was appropriately addressed was the issue of services running on the system. I presented more than one method for determining parts of the puzzle that led to knowing what to recommend for hardening this area.

The areas of writing to logs, monitoring logs, and system backups were addressed. It was unfortunate that I did not have time to write some Perl scripts to address automation of the monitoring and backups during this project. In the absence of existing tools, I usually roll my own script. I find Perl to be particularly effective because it is a versatile scripting language and will run on almost any platform (as long as you invoke the proper Perl executable at the top of your script).

The following areas were either difficult to check or overlooked by my audit:

1. Though my checklist included running a password cracker, I did not do this here, mostly for the practical reason that it was a borrowed system. The password for user *mary* was known to be weak, since it was shown in the example where I proved that the shadow file was using MD5. In practice, I would run either **crack** or **john-the-ripper** to crack the shadow passwords. This is most important because one of the biggest security vulnerabilities is people. Yes, the ordinary users. Most people choose easy-to-remember passwords, but the problem is that most of them are also easy-to-guess. Social engineering is used frequently to find out information about users, which can then be used to guess passwords. In my last job, I wrote a password security policy and documentation for users to be able to make good passwords. It is the power of the cracking part of the regular audit which will find if users are complying with the policy. I find that telling users that this is a serious responsibility usually results in compliance.
2. My audit did not delve very deeply into the area of privileged users and what they are allowed to do on the system. Careful analysis of which privileged users exist and the implications of this need further investigation.
3. Restricting ssh usage by user was not dealt with here. According to the security policy, I needed to allow certain users to ssh into this box but disallow others. If the users would always be coming in from the same IP addresses, that could be configured. However, in

practice, there's no telling where they will be coming from so we can't configure that in hosts.allow. My checklist needs to come up with a method to handle this issue.

4. Specific file permissions, other than a couple of system files, was an area that was not included in the checklists. It would be nice to construct a filesystem tree which includes all of the default system and user files along with their default permissions so that a check could be made whether or not the permissions on the files is correct.
5. What if this is a development system? How with the user/owner of the box bring code trees over? The process would be tedious with ssh/scp. Perhaps nfs needs to run, but sharing of files could be strictly controlled. There needs to be more research on this one to come up with a reasonable solution.
6. The question of the use of sudo vs. allowing setuid executables needs to be more thoroughly investigated. Especially for a workstation, the user would probably find it tedious to use sudo for each command he/she wanted to run as root, when it is a heck of a lot easier to just su. Perhaps the list of setuid files could be cut down. But there's still the big one -- /bin/su itself.
7. RPC services – what's the best for this system, to use them or not? I ended up shutting them down here, but more investigation needs to be done. Read David Hoelzer's paper "RPC: Your Friend or Foe", SANS 2001. (See References section for complete information.)

Future Directions

To improve this audit, I would try to get hold of some tools which were more current; some of these tools aren't supported as well as I would like, mostly because they are freeware. Freeware is great because it's more creative and is less bound by restrictions, but sometimes it lacks the regular support of commercial tools.

I would also write my own scripts to form a "hardening suite". I would have different flavors of hardening available for different types of systems: workstation, server, firewalls, etc. It would be great to have text-based configuration files that the hardening scripts would read so that each audit could be configured and run in less time than manually doing all the checks on my list.

Running services is a large audit area and needs to be better defined. For example, the identd daemon is one that stuck out like a sore thumb in my audit results and I wound up shutting down that service. But the description of that service – it returns the name of the process owning a particular TCP/IP connection – sounds like a good thing. I could not find anything in any of my reference checklists about this particular daemon. There are other services which are also not thoroughly understood by auditors. What I would like to see is more technical details in checklists so that we can all understand why we are doing some of the things we are doing.

Going into the field and finding out what the particular challenges are for system admins would go a long way toward making the audit checklists and procedures more specific and more appropriate for real-world situations. Learning the solutions that people already have come up with would add to the auditing knowledge base. The questions about sudo, file sharing, and RPC's might then be more easily answered.

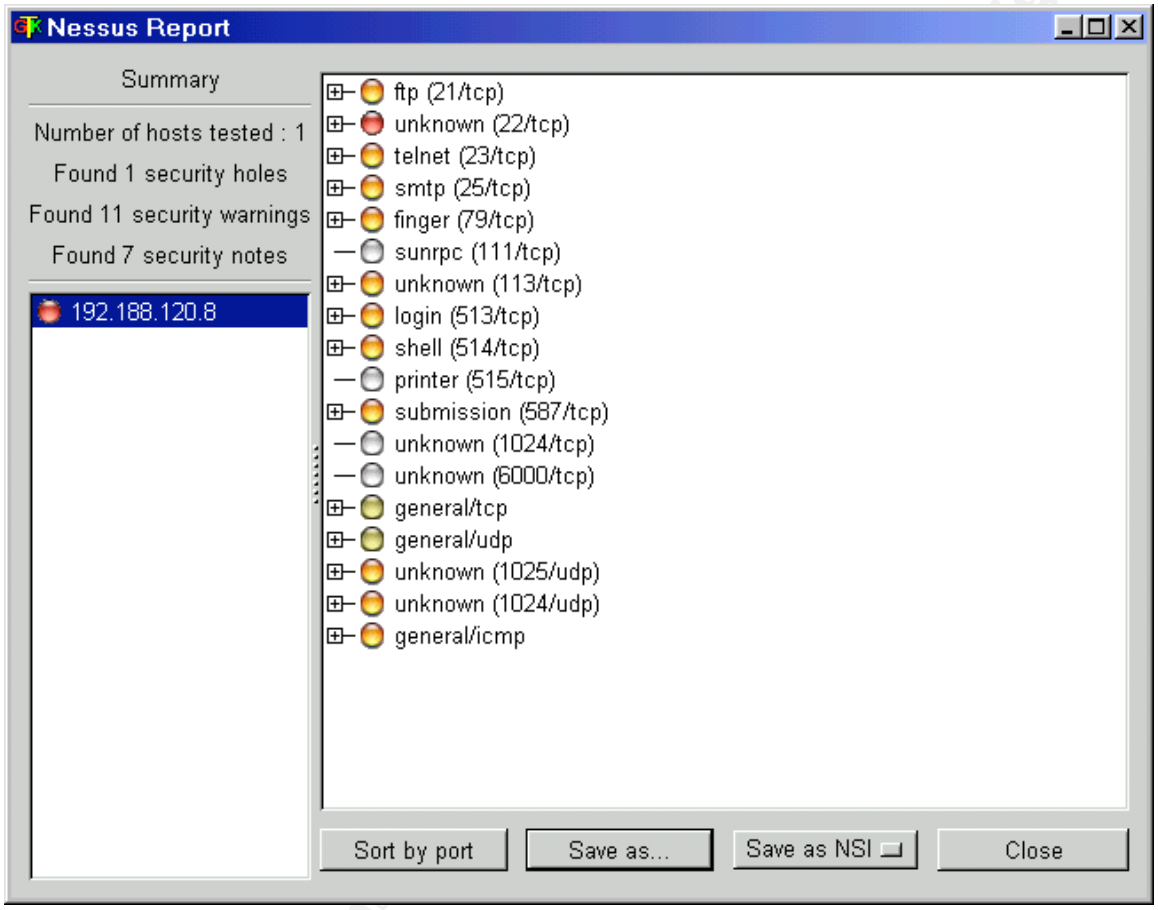
In general, security auditing is a pretty young field. There are still lots of companies who don't perform audits of their systems. They seem to know about financial auditing, but not about security auditing. We need to publish papers (in a wider arena), talk to colleagues, and really evangelize this area of work so that more people know what we do and why it's important. In interviewing for security engineering jobs, I find lots of people who haven't heard of SANS, let alone security auditing. This is very scary.

Thanks

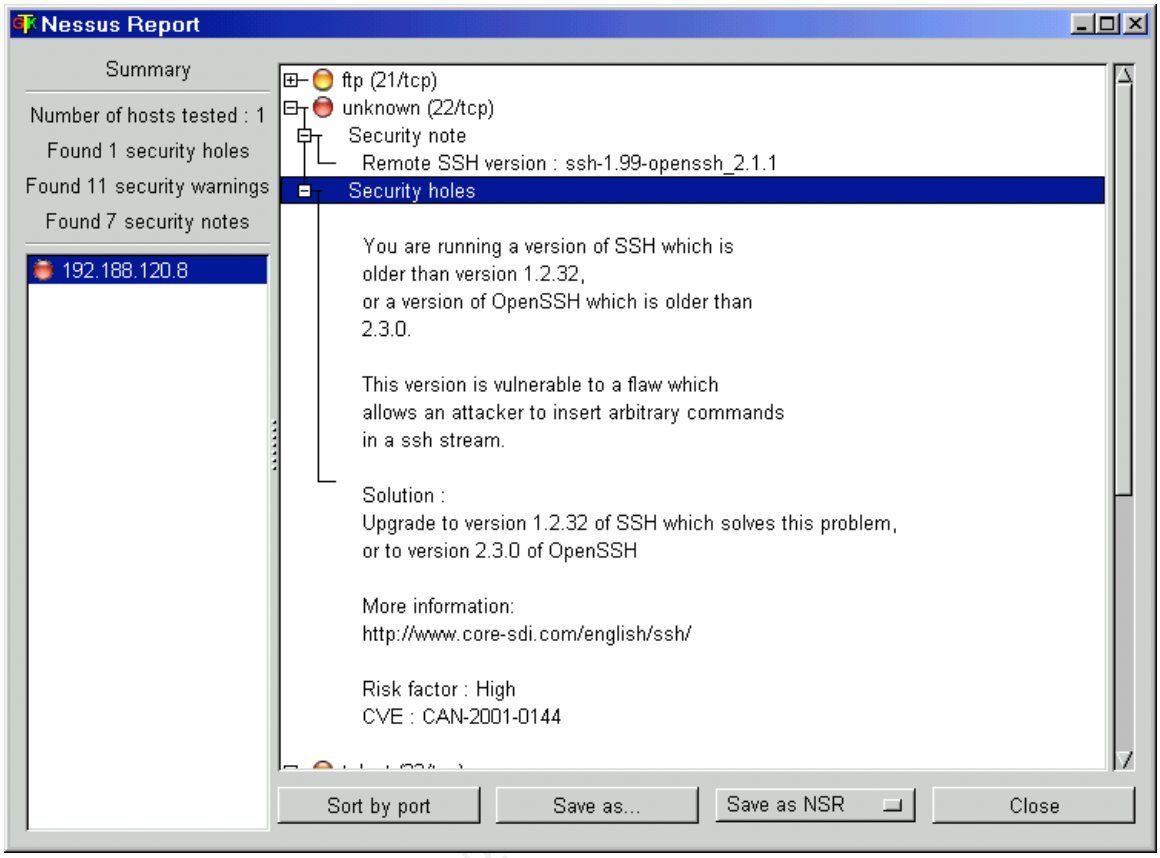
I could not have completed this project were it not for my family and friends, who encouraged me to start and keep working on this project despite motivational problems brought on after being laid off by my .com employer. Thanks especially to Joe, who supplied the small network of systems at his home for my use during this project.

Examples

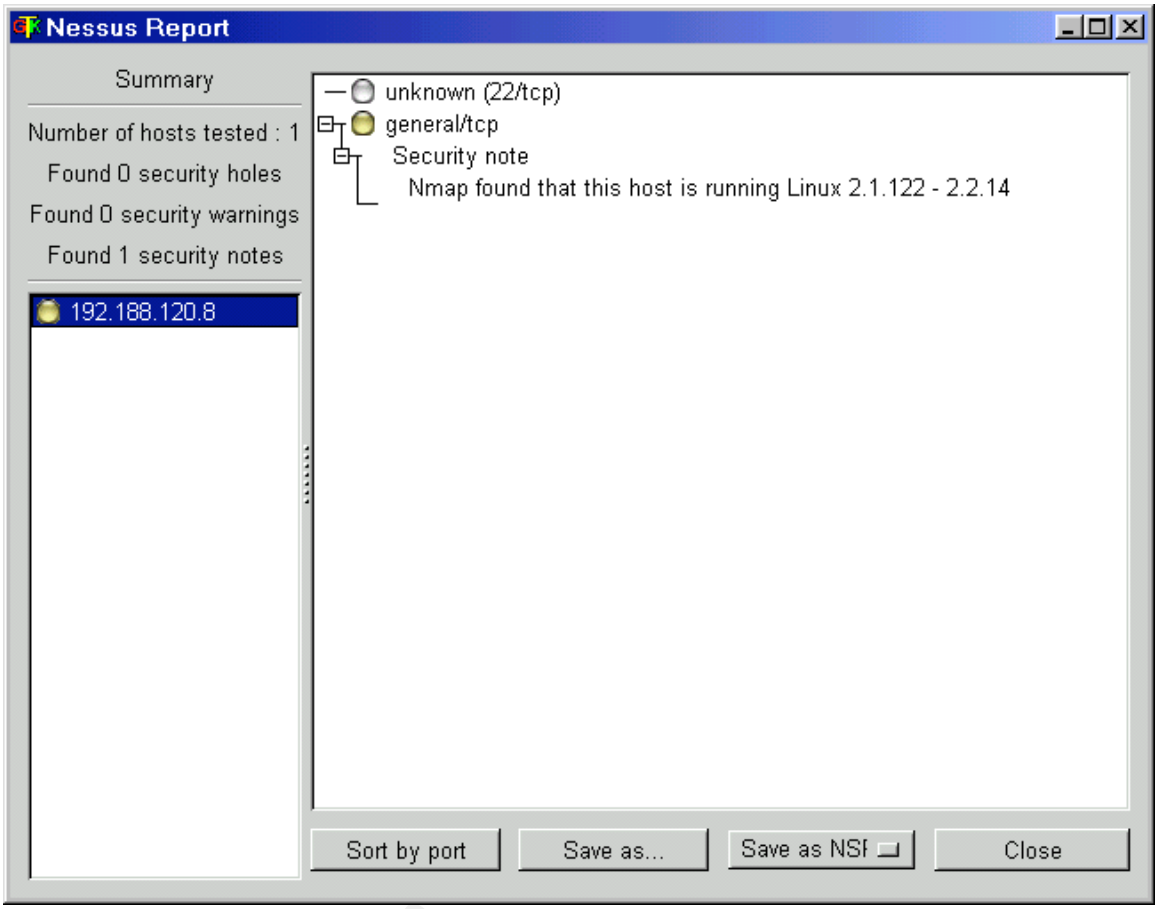
Nessus Example 1



Nessus Example 2



Nessus Example 3



Tools

- Abell, Vic. Purdue University. **Lsof**. URL: ftp: //vic.cc.purdue.edu/pub/tools/unix/lsof/lsof.tar.gz
- Deraison, Renaud. **Nessus** Security Scanner. URL: http://www.nessus.org/download.html
- Farmer, Dan. Computer Oracle and Password System (**COPS**). URL: http://www.fish.com/cops/cops104+.tar.gz
- Fyodor, et. al. Network Mapper (**Nmap**). URL: http://download.insecure.org/nmap/dist/nmap-2.53-1.i386.rpm
- Hansen, Stephen E. and Atkins, Todd. **Swatch**, Automated System Monitoring and Notification. URL: ftp://ftp.stanford.edu/general/security-tools/swatch/swatch.tar.Z.
- Lasser, John and Beale, Jay. "**Bastille** Linux Hardening System 1.2.0". URL: http://bastille-linux.sourceforge.net/
- Source Forge. **Tripwire**. URL: http://www.tripwire.org/files/rpm4/tripwire-2.3-47.i386.tar.gz

References

- Caldera Systems, Inc. “Linux Security Advisories”. URL: http://www.caldera.com/support/security/

- CERT Coordination Center, Carnegie Mellon Software Engineering Institute. “Configure computer operating systems with appropriate object, device, and file access controls.” A practice from the CERT Security Improvement Modules. URL: http://www.cert.org/security-improvement/practices/p070.html.

- Conner, Gary L. The SANS Institute. “Process for Performing, Evaluating and Documenting Host Vulnerability”. May 31, 2001. URL: http://www.sans.org/infosecFAQ/audit/host_vulnerability.htm.

- Green, John and Kolde, Jennifer. The SANS Institute. Advanced Systems Audit and Forensics. SANS 2001 Baltimore, Maryland, May 2001.

- Green, John. The SANS Institute. Auditing Networks with Nmap and Other Tools. SANS 2001 Baltimore, Maryland, May 2001.

- Guardian Digital, Inc. “LinuxSecurity Advisories”. Copyright 2000. URL: http://www.linuxsecurity.com/advisories/redhat.html.

- Hoelzer, David. The SANS Institute. “RPC: Your Friend or Foe”. Version 1.5. SANS 2001, Baltimore, Maryland, May 2001.

- Hsiao, Aron. “Workstation Security Primer”. Copyright 2001 About.com. URL: http://linux.about.com/compute/linux/library/weekly/aa042800c-a1.htm.

- Hughes, David. “Have I Been Hacked?” Sys Admin Magazine, August 2001, Volume 10, Number 8: 30.

- The MITRE Corporation. “Common Vulnerabilities and Exposures (CVE) Reference Map for Source REDHAT”. URL: http://cve.mitre.org/cve/refs/refmap/source-REDHAT.html.

- Naidu, Krishni. The SANS Institute. “Auditing Linux”. Checklist provided to Auditing Networks track attendees at SANS Baltimore, May 2001. URL: http://www.sans.org/checklist/linux_check.htm.

- Negus, Christopher. Red Hat Linux 7 Bible. Foster City: IDG Books Worldwide, Inc., 2001.

- Red Hat, Inc. “Red Hat Linux 7.0 Security Advisories”. Copyright 2001. URL: http://www.redhat.com/support/errata/rh7-errata-security.html.

- Red Hat, Inc. “Online Resources for Red Hat Linux 7”. Copyright 2001. URL: http://www.redhat.com/support/docs/howto/rhl7.html

- Red Hat, Inc. “Red Hat Linux 7.0 The Official Red Hat Linux Reference Guide”. Copyright 2000. URL: http://www.redhat.com/support/manuals/RHL-7-Manual/ref-guide/

- The SANS Institute. Securing Linux Step-by-Step. Copyright 1999, 2000.
- Schneier, Bruce. Secrets & Lies. New York: John Wiley & Sons, Inc., 2000.
- Spitzner, Lance. "Armoring Linux". URL: http://www.linuxnewbie.org/nhf/intel/security/armorlin.html.
- Whelan, Paul. The SANS Institute. "Linux Security Auditing". June 1, 2001. URL: http://www.sans.org/infosecFAQ/audit/linux_sec.htm.
- X_console (shellscope@yahoo.com). "Securing the Home Linux System". URL: http://www.linuxnewbie.org/nhf/intel/security/securehome.html.

Appendix A – List of RedHat 7.0 security update packages and versions (as of 4 July 2001).

(denotes item not applicable to audithost)*

- samba-2.0.10-0.7 *
- samba-common-2.0.10-0.7 *
- samba-client-2.0.10-0.7 *
- Xfree86-3Dlabs-3.3.6-38 *
- Xfree86-8514-3.3.6-38 *
- Xfree86-AGX-3.3.6-38 *
- Xfree86-FBDev-3.3.6-38 *
- Xfree86-Mach32-3.3.6-38 *
- Xfree86-Mach64-3.3.6-38 *
- Xfree86-Mach8-3.3.6-38 *
- Xfree86-Mono-3.3.6-38 *
- Xfree86-P9000-3.3.6-38 *
- Xfree86-S3-3.3.6-38 *
- Xfree86-S3V-3.3.6-38 *
- Xfree86-SVGA-3.3.6-38 *
- Xfree86-VGA16-3.3.6-38 *
- Xfree86-W32-3.3.6-38 *
- LPRng-3.7.4-23
- gnupg-1.0.6-1
- xinetd-2.1.8.9pre15-2
- man-1.5i-4
- krb5-devel-1.2.2-5
- krb5-libs-1.2.2-5
- krb5-workstation-1.2.2-5 *
- minicom-1.83.1-8
- gftp-2.0.8-1
- nfs-utils-0.3.1-7
- mount-2.10r-5
- losetup-2.10r-5
- kernel-2.2.19-7.0.1
- kernel-BOOT-2.2.19-7.0.1 *
- kernel-doc-2.2.19-7.0.1 *
- kernel-ibcs-2.2.19-7.0.1 *
- kernel-pcmcia-cs-2.2.19-7.0.1
- kernel-smp-2.2.19-7.0.1 *
- kernel-source-2.2.19-7.0.1
- kernel-utils-2.2.19-7.0.1
- netscape-common-4.77-1
- netscape-communicator-4.77-1
- netscape-navigator-4.77-1 *
- pine-4.33-7
- ntp-4.0.99k-15 *
- openssh-2.5.2p2-1.7.2
- openssh-clients-2.5.2p2-1.7.2
- openssh-server-2.5.2p2-1.7.2
- openssh-askpass-2.5.2p2-1.7.2
- openssh-askpass-gnome-2.5.2p2-1.7.2
- stunnel-3.10-2
- licq-1.0.2-2 *
- rpm-4.0.2-7x

- rpm-devel-4.0.2-7x
- rpm-build-4.0.2-7x
- rpm-python-4.0.2-7x
- popt-1.6.2-7x
- sgml-tools-1.0.9-9
- slrn-0.9.6.4-0.7
- slrn-pull-0.9.6.4-0.7 *
- mutt-1.2.5i-8.7
- sudo-1.6.3p6-1 *
- joe-2.8-43.7 *
- vixie-cron-3.0.1-61
- xemacs-21.1.14-2.7 *
- xemacs-el-21.1.14-2.7 *
- xemacs-info-21.1.14-2.7 *
- bind-8.2.3-1 *
- bind-devel-8.2.3-1 *
- bind-utils-8.2.3-1
- php-4.0.4pl1-3 *
- php-imap-4.0.4pl1-3 *
- php-ldap-4.0.4pl1-3 *
- php-manual-4.0.4pl1-3 *
- php-mysql-4.0.4pl1-3 *
- php-pgsql-4.0.4pl1-3 *
- mysql-3.23.32-1.7 *
- mysql-devel-3.23.32-1.7 *
- mysql-server-3.23.32-1.7 *
- mysqlclient9-3.23.32-1.7 *
- glibc-2.2-12
- glibc-common-2.2-12 *
- glibc-devel-2.2-12
- glibc-profile-2.2-12 *
- nscd-2.2-12 *
- slocate-2.4-1
- rp-pppoe-2.5-1
- ed-0.2-19
- tcsh-6.10-1
- pam-0.72-37
- nss_ldap-122-1.7 *
- cyrus-sasl-1.5.24-11
- apache-1.3.14-3 *
- apache-devel-1.3.14-3 *
- apache-manual-1.3.14-3 *
- mod_ssl-2.7.1-3 *
- mod_php-4.0.3pl1-1 *
- usermode-1.37-2
- gnorpm-0.95.1-5
- imap-2000-3 *
- imap-devel-2000-3 *
- modutils-2.3.21-1
- ncurses-5.2-2
- ncurses-devel-5.2-2
- ghostscript-5.50-8
- iputils-20001010-1
- tmpwatch-2.6.2-1.7
- esound-0.2.20-1
- esound-devel-0.20-1

Appendix B (Red Hat 7)

CRYPT(3) Library functions CRYPT(3)

NAME

crypt - password and data encryption

SYNOPSIS

```
#define _XOPEN_SOURCE
#include <unistd.h>

char *crypt(const char *key, const char *salt);
```

DESCRIPTION

crypt is the password encryption function. It is based on the Data Encryption Standard algorithm with variations intended (among other things) to discourage use of hard_ ware implementations of a key search.

key is a user's typed password.
salt is a two-character string chosen from the set [a-zA-Z0-9./]. This string is used to perturb the algo_ rithm in one of 4096 different ways.

By taking the lowest 7 bit of each character of the key, a 56-bit key is obtained. This 56-bit key is used to encrypt repeatedly a constant string (usually a string consisting of all zeros). The returned value points to the encrypted password, a series of 13 printable ASCII characters (the first two characters represent the salt itself). The return value points to static data whose content is overwritten by each call.

Warning: The key space consists of 2**56 equal 7.2e16 pos_ sible values. Exhaustive searches of this key space are possible using massively parallel computers. Software, such as crack(1), is available which will search the por_ tion of this key space that is generally used by humans for passwords. Hence, password selection should, at mini_ mum, avoid common words and names. The use of a passwd(1) program that checks for crackable passwords during the selection process is recommended.

The DES algorithm itself has a few quirks which make the use of the crypt(3) interface a very poor choice for any_ thing other than password authentication. If you are planning on using the crypt(3) interface for a cryptogra_ phy project, don't do it: get a good book on encryption and one of the widely available DES libraries.

CONFORMING TO

SVID, X/OPEN, BSD 4.3

SEE ALSO

login(1), passwd(1), encrypt(3), getpass(3), passwd(5)

September 3, 1994 1

Appendix C

CRYPT(3) FreeBSD Library Functions Manual CRYPT(3)

NAME
crypt - Trapdoor encryption

LIBRARY
SYNOPSIS
#include <unistd.h> char *
crypt(const char *key, const char *salt)

const char *
crypt_get_format(void)

int
crypt_set_format(const char *string)

DESCRIPTION
The crypt() function performs password hashing with additional code added to deter key search attempts. Different algorithms can be used to in the hash. Currently these include the NBS Data Encryption Standard (DES), MD5 and Blowfish. The algorithm used will depend upon the format of the Salt (following the Modular Crypt Format (MCF)), if DES and/or Blowfish is installed or not, and whether crypt_set_format() has been called to change the default.

The first argument to crypt is the data to hash (usually a password), in a null-terminated string. The second is the salt, in one of three forms:

Extended	If it begins with an underscore (``_'') then the DES Extended Format is used in interpreting both the key and the salt, as outlined below.
Modular	If it begins with the string ``$digit$'' then the Modular Crypt Format is used, as outlined below.
Traditional	If neither of the above is true, it assumes the Traditional Format, using the entire string as the salt (or the first portion).

All routines are designed to be time-consuming. A brief test on a Pentium 166/MMX shows the DES crypt to do approximately 2640 crypts a CPU second and MD5 to do about 62 crypts a CPU second.

DES Extended Format:

The key is divided into groups of 8 characters (the last group is null-padded) and the low-order 7 bits of each character (56 bits per group) are used to form the DES key as follows: the first group of 56 bits becomes the initial DES key. For each additional group, the XOR of the encryption of the current DES key with itself and the group bits becomes the next DES key.

The salt is a 9-character array consisting of an underscore followed by 4 bytes of iteration count and 4 bytes of salt. These are encoded as printable characters, 6 bits per character, least significant character

first. The values 0 to 63 are encoded as ``./0-9A-Za-z''. This allows 24 bits for both count and salt.

The salt introduces disorder in the DES algorithm in one of 16777216 or 4096 possible ways (ie. with 24 or 12 bits: if bit i of the salt is set, then bits i and i+24 are swapped in the DES E-box output).

The DES key is used to encrypt a 64-bit constant using count iterations of DES. The value returned is a null-terminated string, 20 or 13 bytes (plus null) in length, consisting of the salt followed by the encoded 64-bit encryption.

Modular crypt:

If the salt begins with the string $digit$ then the Modular Crypt Format is used. The digit represents which algorithm is used in encryption. Following the token is the actual salt to use in the encryption. The length of the salt is limited to 16 characters--because the length of the returned output is also limited (_PASSWORD_LEN). The salt must be terminated with the end of the string (NULL) or a dollar sign. Any characters after the dollar sign are ignored.

Currently supported algorithms are:

1. MD5
2. Blowfish

Other crypt formats may be easilly added. An example salt would be:

3thesalt$rest

Traditional crypt:

The algorithm used will depend upon whether crypt_set_format() has been called and whether a global default format has been specified. Unless a global default has been specified or crypt_set_format() has set the format to something else, the built-in default format is used. This is currently DES if it is available, or MD5 if not.

How the salt is used will depend upon the algorithm for the hash. For best results, specify at least two characters of salt.

The crypt_get_format() function returns a constant string that represents the name of the algorithm currently used. Valid values are `des', `blf' and `md5'.

The crypt_set_format() function sets the default encoding format according to the supplied string.

The global default format can be set using the /etc/auth.conf file using the `crypt_format' property.

RETURN VALUES

crypt() returns a pointer to the encrypted value on success, and NULL on failure. Note: this is not a standard behaviour, AT&T crypt() will always return a pointer to a string.

crypt_set_format() will return 1 if the supplied encoding format was valid. Otherwise, a value of 0 is returned.

SEE ALSO

login(1), passwd(1), auth_getval(3), cipher(3), getpass(3), auth.conf(5), passwd(5)

BUGS

The crypt() function returns a pointer to static data, and subsequent calls to crypt() will modify the same data. Likewise, crypt_set_format() modifies static data.

HISTORY

A rotor-based crypt() function appeared in Version 6 AT&T UNIX. The current style crypt() first appeared in Version 7 AT&T UNIX.

The DES section of the code (FreeSec 1.0) was developed outside the United States of America as an unencumbered replacement for the U.S.-only NetBSD libcrypt encryption library. Users should be aware that this code (and programs staticly linked with it) may not be exported from the U.S., although it apparently can be imported.

AUTHORS

-nosplit Originally written by
David Burren <davidb@werj.com.au>, later additions and changes by
Poul-Henning Kamp,
Mark R V Murray,
Kris Kennaway,
Brian Feldman,
Paul Herman and
Niels Provos.

BSD January 19, 1997 3

```
/*
 * Copyright (c) 1999
 *     Mark Murray.  All rights reserved.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 * 1. Redistributions of source code must retain the above copyright
 *    notice, this list of conditions and the following disclaimer.
 * 2. Redistributions in binary form must reproduce the above copyright
 *    notice, this list of conditions and the following disclaimer in the
 *    documentation and/or other materials provided with the distribution.
 *
 * THIS SOFTWARE IS PROVIDED BY MARK MURRAY AND CONTRIBUTORS ``AS IS'' AND
 * ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
 * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
 * ARE DISCLAIMED.  IN NO EVENT SHALL MARK MURRAY OR CONTRIBUTORS BE LIABLE
 * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
 * DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
 * OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
 * HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
```

```
 * LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
 * OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
 * SUCH DAMAGE.
 *
 * $FreeBSD: src/lib/libcrypt/crypt.c,v 1.19 2001/03/11 16:05:43 markm Exp $
 *
 */

#if defined(LIBC_SCCS) && !defined(lint)
static const char rcsid[] =
"$FreeBSD: src/lib/libcrypt/crypt.c,v 1.19 2001/03/11 16:05:43 markm Exp $";
#endif /* LIBC_SCCS and not lint */

#include <sys/types.h>
#include <string.h>
#include <libutil.h>
#include "crypt.h"

static const struct {
        const char *const name;
        char *(*const func)(const char *, const char *);
        const char *const magic;
} crypt_types[] = {
#ifdef HAS_DES
        {
                "des",
                crypt_des,
                NULL
        },
#endif
        {
                "md5",
                crypt_md5,
                "$1$"
        },
#ifdef HAS_BLOWFISH
        {
                "blf",
                crypt_blowfish,
                "$2"
        },
#endif
        {
                NULL,
                NULL
        }
};

static int crypt_type = -1;

static void
crypt_setdefault(void)
{
        char *def;
        int i;
```

```
	if (crypt_type != -1)
		return;
	def = auth_getval("crypt_default");
	if (def == NULL) {
		crypt_type = 0;
		return;
	}
	for (i = 0; i < sizeof(crypt_types) / sizeof(crypt_types[0]) - 1; i++) {
		if (strcmp(def, crypt_types[i].name) == 0) {
			crypt_type = i;
			return;
		}
	}
	crypt_type = 0;
}

const char *
crypt_get_format(void)
{

	crypt_setdefault();
	return (crypt_types[crypt_type].name);
}

int
crypt_set_format(char *type)
{
	int i;

	crypt_setdefault();
	for (i = 0; i < sizeof(crypt_types) / sizeof(crypt_types[0]) - 1; i++) {
		if (strcmp(type, crypt_types[i].name) == 0) {
			crypt_type = i;
			return (1);
		}
	}
	return (0);
}

char *
crypt(char *passwd, char *salt)
{
	int i;

	crypt_setdefault();
	for (i = 0; i < sizeof(crypt_types) / sizeof(crypt_types[0]) - 1; i++) {
		if (crypt_types[i].magic != NULL && strncmp(salt,
		   crypt_types[i].magic, strlen(crypt_types[i].magic)) == 0)
			return (crypt_types[i].func(passwd, salt));
	}
	return (crypt_types[crypt_type].func(passwd, salt));
}
```

Last Updated: July 24th, 2017

Upcoming Training

SANS Network Security 2017	**Las Vegas, NV**	**Sep 10, 2017 - Sep 17, 2017**	**Live Event**
SANS AUD507 (GSNA) @ Canberra 2017	**Canberra, Australia**	**Oct 09, 2017 - Oct 14, 2017**	**Live Event**
SANS OnDemand	**Online**	**Anytime**	**Self Paced**
SANS SelfStudy	**Books & MP3s Only**	**Anytime**	**Self Paced**